

AN ONLINE MANIFOLD LEARNING APPROACH FOR MODEL REDUCTION OF DYNAMICAL SYSTEMS*

LIQIAN PENG[†] AND KAMRAN MOHSENI[‡]

Abstract. In this article, an online manifold learning approach, implicitly reduced Picard iteration (IRPI), is developed for dimensional reduction of dynamical systems. The approach may be viewed as a variant of Picard iteration combined with a model reduction procedure. Specifically, starting with a test solution, we solve a reduced model to obtain a better approximation, and repeat this process until sufficient accuracy is achieved. The reduced model is obtained by projecting the full model onto a subspace which is spanned by the dominant modes of an extended data ensemble. The extended data ensemble in this article contains not only the state of some snapshots of the approximating solution from the previous iteration, but also the associated vector field. Therefore, the proposed manifold learning procedure takes advantage of the information of the original dynamical system to reduce also the dynamics. Moreover, it is computed in the online stage, as opposed to offline learning in many projection-based model reduction techniques which needs prior calculations or experiments. It has been proved that the proposed sequence of functions converges to the actual solution of the original system as long as the vector field of full model is locally Lipschitz on an open set that contains the solution trajectory. Good accuracy of the proposed technique was demonstrated in several numerical examples from linear convection-diffusion equation to nonlinear Burgers equation. In order to save more computational cost, the IRPI technique is extended to a local model reduction approach by partitioning the entire time domain into several subintervals and obtaining a series of local reduced models with much lower dimensions. The accuracy and efficiency of the locally reduced model are shown through the numerical simulation of several problems including the Navier-Stokes equation in a lid-driven cavity flow problem.

Key words. online, manifold learning, Picard iteration, model reduction, locally linear projection

AMS subject classifications. 78M34, 37M99, 65L99, 34C40, 74H15, 37N10

1. Introduction. Reducing statistical data redundancy in high dimensional problems could enable a low dimensional reduction without a loss of dominant information contents. The problem of manifold learning and dimensionality reduction appears in many fields in science and engineering including pattern recognition and machine learning [11], signal analysis [1], image processing [34], electrical power grids [28, 30], structural dynamics [2], and chemical reaction systems [17, 39] to list a few. The classical linear techniques for manifold learning such as principal component analysis (PCA) [25, 21] and multidimensional scaling [23] are efficient to discover the true structure of data lying on or near a linear subspace of the input space. In order to discover the nonlinear structures, many prominent nonlinear algorithms are proposed, such as Isomap [40], locally linear embedding [35] and Laplacian eigenmaps [9].

In many applications of dynamical systems, conventional methods of direct numerical simulation of all scales in a system are often computationally intensive and they require massive computing resources. For this reason, during the past several decades, many efforts have been put forward to develop reduced models on large-scale complex systems. The core of model reduction is to provide an efficient computational

* This research was supported by the Office of Naval Research.

[†]Department of Mechanical and Aerospace Engineering, and Institute for Networked Autonomous Systems, University of Florida, Gainesville, FL, 32611-6250 (liqianpeng@ufl.edu).

[‡]Department of Mechanical and Aerospace Engineering, Department of Electrical and Computer Engineering, and Institute for Networked Autonomous Systems, University of Florida, Gainesville, FL, 32611-6250 (mohseni@ufl.edu).

prototyping tool to replace a high order system of differential equations with a system of substantially lower dimension, whereby only the most dominant properties of the full system are preserved. Several model reduction techniques have been studied in various fields. Krylov subspace techniques [6, 7] extend the success of extracting accurate reduced models from linear systems to nonlinear ones. Method of balanced truncation [42, 38, 22] has been widely studied by the control community. Proper orthogonal decomposition (POD, also known as Karhunen-Loève decomposition, singular systems analysis, singular value decomposition and principle component analysis) [24] is a well-know method which has a wide application in CFD-based modelling and control [20, 36, 5, 10]. Although POD, as a linear model reduction technique, always looks for linear subspace instead of its curved submanifold, it is computationally tractable and able to capture the dominant patterns in a nonlinear dynamical system. A typical POD procedure involves an offline-online strategy. In the offline manifold learning stage, the standard singular value decomposition (SVD) is carried out to find a set of empirical eigenfunctions (also known as reduced order bases, empirical basis functions, and empirical orthogonal functions). The data vectors, collected from experiments or numerical simulations of the full systems, approximately rely on a linear (or an affine) subspace spanned by empirical eigenfunctions with least mean square error for a specified dimension. In the online stage a reduced model obtained by the Galerkin projection is computed to obtain the solution. Recently, many new model reduction techniques based on POD were developed to reduce the complexity of evaluating the nonlinear term through the standard POD-Galerkin approach, such as missing point estimation [4], trajectory piecewise-linear approximation [32, 33], empirical interpolation method [8, 18], and discrete empirical interpolation [12].

Although the offline manifold learning approach can save more computational time for the online computation, it suffers from two main disadvantages. First, the construction of empirical eigenfunctions during the offline stage is expensive, as it requires experimental results or solutions of a number of large-scale systems in the high dimensional space. Secondly, there is no guarantee that the solutions associated with a different set of initial conditions or parameters rely on the same subspace spanned by prior empirical data vectors. Therefore, the classical POD method usually lacks robustness with respect to parameter change and can yield unpredictable results [31]. For these reasons, an efficient reduced model with high fidelity based on online manifold learning is preferable.¹ However, much less effort has been spent in the field of model reduction via online manifold learning. In [26, 29] solution vectors from previous time steps are extracted to reveal the spatial patterns at current step. Updated empirical eigenfunctions are computed immediately after the reduced-order equations are advanced for one step. Yet this method might not be feasible for nonlinear systems if snapshots in future steps leave the subspace spanned by solution vectors of pervious steps. In [30] dynamic iteration using reduced model combines the idea of waveform relaxation technique and model reduction, which simulates each subsystem connected to model reduced versions of the other subsystems. Even without any prior empirical data, this method can effectively reduce the dimension of large scale molecular systems to obtain a convergent solution.

Both online learning and offline learning techniques invariably involve a trade-off between complexity and accuracy/reliability of the model. Although the former has better descriptive or predictive features, one usually resorts to a more complex

¹To avoid confusion, “online” in this article means that the manifold learning is carried out in the online stage, as opposed to the offline stage.

reduced model which may be even as expensive as to solve the full system. The balance between complexity and accuracy of reduced models raises three problems: 1) Without prior empirical data, how can one find a subspace where the actual solution of a nonlinear system resides? 2) If a trial solution is given initially, how can one tests its accuracy? 3) Is it possible to formulate a fast online learning algorithm for model reduction which can achieve large speedups while maintaining high accuracy?

To answer the first problem, a new framework of iterative manifold learning, implicitly reduced Picard iteration (IRPI), is proposed here for reduced modelling of high-order nonlinear dynamical systems. Similar to the well-known Picard iteration, a trial solution is set at the very beginning. In each iteration, a set of updated empirical eigenfunctions are constructed by extracting dominant modes from an extended data ensemble, then a more accurate solution is obtained by solving the reduced equation on a new subspace spanned by these empirical eigenfunctions. The extended data ensemble not only contains the state vectors of some snapshots of the solution in the previous iteration, but also the associated vector field. Therefore, the manifold learning process essentially takes advantage of the information from the original dynamical system. One could argue that this results in actively reducing the dynamics as well as the solution snapshots. In principle, any manifold learning techniques could be used to find a set of empirical functions of a subspace, and any projection methods can produce a reduced model. In this article, we take the standard POD-Galerkin approach for example. Both analytical results and numerical simulations indicate that a sequence of functions asymptotically converges to the solution of the full system.

The above mentioned approach can also solve the second problem with a minor modification. If a test solution is given, a linear subspace can be constructed by extracting information from snapshots of the higher dimensional space. A posterior error estimation is given by the difference between the trial solution and the better solution obtained by the reduced model in the subspace.

For the third problem, we seek an efficient online manifold learning approach with significant speedups. In many nonlinear problems, the solutions have significant variations as time proceeds. Accordingly, the relevant reduced model need calculation with an increasing number of modes in order to capture the main properties of the entire trajectory. To this effect, a local model reduction technique based on IRPI, locally linear projection (LLP), is proposed where the whole entire domain is partitioned into several subintervals. The full model is, then, projected onto a set of local subspaces with significantly lower dimensions. The locally reduced model is updated when the solution moves from one subinterval to another. The efficiency of this method is illustrated in a numerical example: the two-dimensional lid-driven cavity flow problem.

The remainder of this article is organized as follows. Since the model reduction techniques in this paper fall in the category of projection methods, the classical Galerkin-POD approach and its ability to minimize truncation error is briefly reviewed in section 2. Then a review of Picard iteration is presented in section 3 and for introduction to a reduced Picard iteration technique. A memory-efficient and iterative manifold learning technique is devolved in section 4, which is a modified version of the reduced Picard iteration. After presenting the algorithm in section 4.1, we provide analytical results of convergence in section 4.2 and a complexity analysis in section 4.3. We also illustrate that the proposed method has a high accuracy for linear convection-diffusion equation and nonlinear Burgers equation in section 4.4. In section 5, the locally linear projection approach is proposed to save more computa-

tional cost. The performance of this technique is evaluated in a lid-driven cavity flow problem. Finally, conclusions are offered in section 6.

2. Background on Model Reduction Techniques. In physical world, the description of many dynamical systems involves solving partial differential equations (PDEs) with first order time-derivative terms. These include linear heat equation, Schrodinger equation, Burgers equation, and Navier-Stokes equations to name a few. After spatial discretization (for example, using finite difference or finite element methods), the full system projected on \mathbb{R}^n is given by a set of ordinary differential equations (ODE) and its initial condition,

$$(2.1) \quad \dot{x} = f(t, x); \quad x(0) = x_0.$$

Here $x : [0, T] \rightarrow \mathbb{R}^n$ and the discretized vector field $f : [0, T] \times \mathbb{R}^n \rightarrow \mathbb{R}^n$ is an algebraic operator. To indicate the dependence of x on its initial value, let $w(t; x_0) := x(t)$ be a function of both t and x_0 . The evolution rule can be considered as a mapping from the phase space to itself which is parameterized by time. By definition, $w(t; x_0)$ is a flow which gives an orbit in \mathbb{R}^n as t varies over $[0, T]$ for a fixed x_0 . The orbit contains a sequence of states (solution vectors) that follow from x_0 .

If f is a nonlinear operator and the dimension n is large, lower dimensional models that approximate the full system are often beneficial in terms of computational saving. As a classical projection-based model reduction method, the Galerkin-POD method is briefly reviewed here, for its capability of providing an accurate reduced model with significantly lower dimension, also because of its application in our online manifold learning approach for model reduction.

2.1. Galerkin Projection. Galerkin projection can provide a lower dimensional approximation by projecting the full system onto a linear (or an affine) subspace. For any n dimensional linear subspace S , there exists an $n \times k$ orthonormal matrix $\Phi = (\phi_1, \dots, \phi_k)$, whose columns form a complete basis associated with it. The orthonormality of the column matrix requires that $\Phi^T \Phi = I_k$ where I_k is the $k \times k$ identity matrix. Any state $x \in \mathbb{R}^n$ can be projected onto the subspace S by a linear projection mapping,

$$(2.2) \quad z = \Phi^T x$$

where $z \in \mathbb{R}^k$ is the coordinate in the subspace coordinate system and the superscript T denotes matrix transpose. In the original coordinate system, this point can be expressed as

$$(2.3) \quad \tilde{x} = \Phi z.$$

Substituting (2.2) into (2.3), we immediately obtain the projection $\tilde{x} \in S$ of a point x in the original coordinate system, i.e.,

$$(2.4) \quad \tilde{x} = P x,$$

where $P = \Phi \Phi^T \in \mathbb{R}^{n \times n}$. Suppose a vector field $f_k(t, x)$ in the subspace S is constructed by $f_k(t, x) = \Phi^T f(t, \Phi z)$, then a reduced model of the original system (2.1) can be obtained,

$$(2.5) \quad \dot{z} = \Phi^T f(t, \Phi z); \quad z_0 = \Phi^T x_0.$$

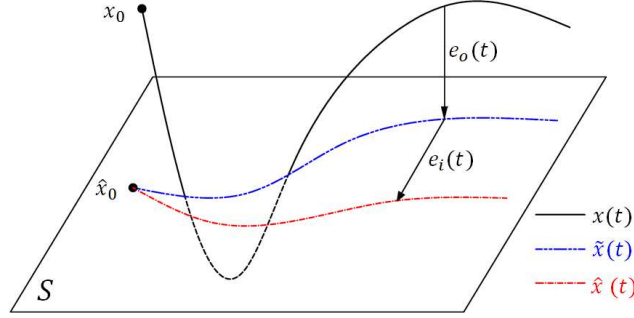


FIGURE 2.1. (Color online.) Illustration of the actual solution $x(t)$ for the original system (2.1), the projected solution $\tilde{x}(t)$ on the subspace S (2.4) and the approximating solution $\hat{x}(t)$ computed by the reduced model (2.6). The component of error parallel to S is given by $e_i(t) = \hat{x}(t) - \tilde{x}(t)$, and the component of error orthogonal to S is given by $e_o(t) := \tilde{x}(t) - x(t)$. Reproduced from [31].

Hence, the reduced solution in the original coordinate system $\hat{x}(t) = \Phi z(t) \in \mathbb{R}^n$ is equivalent to the following initial value problem

$$(2.6) \quad \dot{\hat{x}} = Pf(t, \hat{x}); \quad \hat{x}_0 = Px_0.$$

The error of the reduced model formed by Galerkin projection can be defined as $e(t) := \hat{x}(t) - x(t)$. Let $e_o(t) := (I - P)e(t)$, which denotes the error component orthogonal to the projected subspace S , and $e_i(t) := Pe(t)$, which denotes the component of error parallel to S . Thus $e_o(t)$ and $e_i(t)$ are orthogonal to each other. $e_o(t)$ is the error between $x(t)$ and its projection $\tilde{x}(t)$ onto S , which comes from POD or other dimensionality reduction techniques. But since the system is evolutionary with time, further approximations of projection based reduced model result in an additional error $e_i(t)$ (see. Figure 2.1) [31].

We can also project the full model onto an affine subspace by shifting S by \bar{x} , where \bar{x} is the mean value of x over a given time interval $[0, T]$, and given by

$$(2.7) \quad \bar{x} = \frac{1}{T} \int_0^T x(t) dt.$$

In principle, there is no fundamental difference between a linear projection and an affine projection. (2.2) through (2.6) are still valid if we apply the following substitutions,

$$(2.8) \quad x \rightarrow \begin{pmatrix} x \\ 1 \end{pmatrix}, \quad z \rightarrow \begin{pmatrix} z \\ 1 \end{pmatrix}, \quad \Phi \rightarrow \begin{pmatrix} \Phi & \bar{x} \\ 0 & 1 \end{pmatrix}, \quad \Phi^T \rightarrow \begin{pmatrix} \Phi^T & -\bar{z} \\ 0 & 1 \end{pmatrix}$$

where $\bar{z} = \Phi^T \bar{x}$. Similar transformation rules can be applied for \hat{x} and \tilde{x} .

Although a reduced model can be constructed by Galerkin projection on an arbitrary subspace, its solution might not approximate the original one with a high accuracy unless the subspace is (approximately) invariant of the solution $w(t, x_0)$. In this article, a subspace S^* called is *invariant* of $w(t, x_0)$ (or invariant, for short) if $x_0 \in S^*$, and the solution orbit given by $w(t, x_0) = x(t)$ lies on S^* ,

$$(2.9) \quad P^*x = x,$$

where $P^* \in \mathbb{R}^{n \times n}$ is the projection operator corresponding to S^* . In this case, the governing equation of the reduced model (2.6) and the full model (2.1) must be identical, which requires P^* satisfies the following condition, i.e.,

$$(2.10) \quad (P^*)f(t, x) = f(t, x).$$

Equations (2.9) and (2.10) means an invariant projection operator P^* not only preserve the state vector, but also preserve the vector field, i.e. the dynamics. Since an invariant subspace S^* is uniquely determined by its basis vectors, the next subsection discuss the classical POD method which can construct empirical eigenfunctions from a data ensemble.

2.2. Proper Orthogonal Decomposition. The idea of POD is to deliver a set of empirical eigenfunctions that the error for representing the given data onto the spanned subspace is in a certain least squares optimal sense [20]. Suppose a data ensemble is given by $x(t) \in \mathbb{R}^n$ with $t \in [0, T]$. POD seeks a projection P of fixed dimension k that minimizes the total error [20]

$$(2.11) \quad \int_0^T \|x(t) - Px(t)\|^2 dt.$$

After defining an $n \times n$ autocorrelation matrix

$$(2.12) \quad R := \int_0^T x(t)x(t)^T dt,$$

a necessary condition to minimize the error (2.11) is to solve the eigenvalue problem

$$(2.13) \quad R\phi_i = \lambda_i\phi_i, \quad \lambda_1 \geq \dots \geq \lambda_n \geq 0,$$

and construct the optimal projection operator $P \in \mathbb{R}^{n \times n}$ through

$$(2.14) \quad P = \sum_{i=1}^k \phi_i \phi_i^T.$$

In the numerical simulation, it is more feasible to rewrite the entire data ensemble $x(t)$ in a discretized form,

$$(2.15) \quad X := [x(t_1)\sqrt{\delta_1}, \dots, x(t_m)\sqrt{\delta_m}],$$

where t_1, \dots, t_m are discrete temporal points, and $\delta_1, \dots, \delta_m$ are quadrature coefficients. In this case, the integral in (2.12) may be written as $R = XX^T$, and the empirical eigenfunctions $\{\phi_i\}_{i=1}^n$ can be computed by (2.13). However, when the number of snapshots m is smaller than the dimension of states n , it is more efficient to solve the following $n \times m$ singular value decomposition (SVD) problem,

$$(2.16) \quad X = U\Lambda^{-1/2}V^T,$$

where $U \in \mathbb{R}^{n \times r}$ and $V \in \mathbb{R}^{m \times r}$ are orthogonal, and $\Lambda \in \mathbb{R}^{r \times r}$ is a diagonal matrix which consists of $r = \min(n, m)$ nonnegative diagonal elements arranged in decreasing order, i.e. $\lambda_1 \geq \dots \geq \lambda_r \geq 0$. These singular values in Λ are the same as in (2.13). If the first k singular values are significantly larger than the rest, one can take a good approximation of X by only calculating k column vectors of U and V corresponding

to the k largest singular values. This can be much quicker and more efficient if $k \ll r$. Let u_i be the i th column of U , then the POD basis calculated by the truncated SVD is given by

$$(2.17) \quad \Phi := [u_1, \dots, u_k] \in \mathbb{R}^{n \times k}.$$

Although the truncated SVD is no longer an exact decomposition of the original matrix X , but it provides the best approximation to the original data X with least square error for the same dimension. In the rest of this paper, a SVD process refers to truncated SVD unless specified otherwise.

Let E_r denotes the energy of the full system,

$$(2.18) \quad E_r = \int_0^T \|x(t)\|^2 dt = \sum_{i=1}^r \lambda_i,$$

and E_k denotes the energy in the optimal k dimensional subspace,

$$(2.19) \quad E_k = \int_0^T \|Px(t)\|^2 dt = \sum_{i=1}^k \lambda_i.$$

A criterion can be set to limit the approximation error in energy by a certain fraction η . Then, we seek $k \ll r$ so that

$$(2.20) \quad \frac{E_k}{E_r} > \eta,$$

where k represents the number of empirical eigenfunctions necessary for an approximation with relative mean square error less than $1 - \eta$. The POD basis is optimal for model reduction since no other linear basis can represent better energy approximation in the same dimension.

The key idea of POD and other projection based reduced models is to find an (approximating) invariant subspace on which all the solution vectors live. Fortunately, for most large scale complex systems although the states are in general represented by a high dimensional space, the solution orbit is actually embedded in an invariant subspace with significantly lower dimension. Therefore, it is possible to construct an accurate reduced model by forming a projection. In the past decades, many offline manifold learning techniques are developed to approximate the invariant subspace through a pre-computed data ensemble [3]. Although these methods can significantly increase the computational speed during the online stage, the cost of construction of data ensemble in the offline stage is often very expensive. Moreover, there is no guarantee that the pre-computed data vectors can span an invariant subspace for the same dynamical system but with different parameters. For these reasons, it is desired to develop an online manifold learning technique with inexpensive cost. As shown in the next section, Picard iteration can be employed to obtain POD-required snapshots from the data in the previous iteration. Consequently, it is possible to construct an approximating invariant subspace without a precalculated data ensemble obtained in the offline stage, as shown by the reduced Picard iteration (RPI) approach.

3. Reduced Picard Iteration (RPI) Approach. In this section, we begin with a brief review of the well-known Picard iteration process for solving the initial value problem. Subsequently, the RPI approach is introduced by combining Picard iteration with some model reduction techniques.

3.1. Picard Iteration. Picard iteration is a constructive procedure for establishing the existence and uniqueness of a solution to an initial value problems. Formally integrating the equation in (2.1) with respect to t yields

$$(3.1) \quad x(t) = x_0 + \int_0^t f(\tau, x(\tau)) \, d\tau.$$

The right side of (3.1) can be viewed as an operator, T , acting on the function $x(t)$

$$(3.2) \quad T(x) = x_0 + \int_0^t f(\tau, x(\tau)) \, d\tau.$$

Picard iteration process starts with selecting a test function $x^0(t)$. Then, the operator T is applied to this test function to obtain a *better* approximation, $x^1 = T(x^0)$. Recursive application of this operator generates a sequence of functions

$$(3.3) \quad x^j(t) = T(x^{j-1})(t), \quad j = 1, 2, \dots,$$

where the superscript j denotes the j th iteration. The following theorem summarize existence and uniqueness of solutions to first-order equations with given initial conditions.

THEOREM 3.1 (Picard-Lindelöf Existence and Uniqueness [27]). *Suppose there is a closed ball of radius b around a point $x_0 \in \mathbb{R}^n$ such that $f : J_0 \times B_b(x_0) \rightarrow \mathbb{R}^n$ is a uniformly Lipschitz function of $x \in B_b(x_0)$ with constant K , and a continuous function of t on $J_0 = [-a, a]$. Then the initial value problem (2.1) has a unique solution $x(t)$ for $t \in J_0$, provided that $a = b/M$ where*

$$M = \max_{x \in B_b(x_0), t \in J_0} \|f(t, x)\|.$$

In the study of differential equations, this theorem is an important application of the the Banach fixed point theorem to a sequence of functions defined by Picard iteration. The fixed point $x^* = T(x^*)$ is proved to be the solution to the initial value problem (2.1). The Picard iteration technique (algorithm 1) is reproduced here for comparison with algorithms in the following sections, and also because our methods use ideas from Picard iteration.

Algorithm 1 Picard iteration

Require: The initial value problem (2.1).

Ensure: An approximating solution $\hat{x}(t)$.

Set a test function $\hat{x}^0(t)$ as the trial solution. Initialize the iteration number $j = 0$.

repeat

1: Update the iteration number $j = j + 1$.

2: Compute $x^j(t) = T(x^{j-1})(t)$ (3.2) to update the approximating solution.

until $\|x^j - x^{j-1}\|_\infty < \epsilon$, where ϵ is the tolerant error.

Obtain the final solution $\hat{x}(t) = \hat{x}^j(t)$.

3.2. Reduced Picard Iteration. Even though Picard iteration has good asymptotic convergence properties, it has little practical value for solving initial value problems because recursive time integrations in a high dimensional space are very expensive. Therefore, it is desired to combine model reduction techniques and Picard iteration for cost saving purposes. After rewriting (3.3) in a differential form,

$$(3.4) \quad \dot{x}^j = f(t, x^{j-1}); \quad x^j(0) = x_0,$$

we can form a projection of (3.4) and obtain a reduced model in the original coordinate system,

$$(3.5) \quad \dot{\hat{x}}^j = P^j f(t, P^j \hat{x}^{j-1}); \quad \hat{x}_0^j = P^j(x_0).$$

The associated reduced model in the subspace coordinate system is

$$(3.6) \quad \dot{z}^j = (\Phi^j)^T f(t, \Phi^j \tilde{z}^{j-1}); \quad z_0^j = \Phi^j(x_0),$$

with a solution

$$(3.7) \quad z^j(t) = z_0^j + (\Phi^j)^T \int_0^t f(\tau, \Phi^j \tilde{z}^{j-1}(\tau)) d\tau,$$

where \tilde{z}^{j-1} is the transformed coordinate of z^{j-1} from S^{j-1} to S^j ,

$$(3.8) \quad \tilde{z}^{j-1} = (\Phi^j)^T(\tilde{y}^{j-1}) = (\Phi^j)^T \Phi^{j-1}(z^{j-1}).$$

Equations (3.4), (3.6), and (3.5) respectively correspond to (2.1), (2.5), and (2.6). S^j is an approximation for invariant space S^* at iteration j . Parallel to (2.9) and (2.10), we have

$$(3.9) \quad P^j(\hat{x}^{j-1}) = \hat{x}^{j-1},$$

and

$$(3.10) \quad P^j(f(t, \hat{x}^{j-1})) = f(t, \hat{x}^{j-1}).$$

Therefore, both the state vector \hat{x}^{j-1} and the vector field $f(t, \hat{x}^{j-1})$ are invariant under P^j . Let J_m be the maximal open interval containing $t = 0$ while a unique solution exists for the original system (2.1), J_m^j be the corresponding interval for the reduced model, (3.5) and (3.6), at iteration j . $J := J_m \cap [0, T]$, and $J^j := J_m^j \cap [0, T]$. S^j can be expressed as

$$(3.11) \quad S^j = \text{span}(X^{j-1}, F^{j-1}),$$

where

$$(3.12) \quad X^j := \{\hat{x}^j(t) : t \in J^j\},$$

is the approximating orbit at iteration j , and

$$(3.13) \quad F^j := \{f(t, \hat{x}^j(t)) : t \in J^j\},$$

is the associated vector field along the approximating solution.

If the solution orbit is given as “snapshots” $x(t_i)$ at discrete times t_1, \dots, t_m , then one can assemble the data into an $n \times m$ matrix

$$(3.14) \quad \hat{X}^j := [\hat{x}^j(t_1)\sqrt{\delta_1}, \dots, \hat{x}^j(t_m)\sqrt{\delta_m}].$$

Accordingly, samples of vector field along the approximating orbit can form another $n \times m$ matrix,

$$(3.15) \quad \hat{F}^j := [f(t_1, \hat{x}^j(t_1))\sqrt{\delta_1}, \dots, f(t_m, \hat{x}^j(t_m))\sqrt{\delta_m}].$$

A combination of \hat{X}^j and \hat{F}^j gives the *information matrix*, which is used to represent an extended data ensemble

$$(3.16) \quad \hat{Y}^j := [\hat{X}^j, \gamma \hat{F}^j],$$

where γ is a weighting coefficient. $\gamma = 1$ is a typical value to balance the truncation error of \hat{X}^j and \hat{F}^j . By (3.11), we have

$$(3.17) \quad S^j \approx \text{span}(\hat{Y}^{j-1}),$$

which means the basis vectors of S^j can be obtained by SVD of \hat{Y}^{j-1} . However, it should be emphasized that the POD (or SVD) method is not the unique method to construct the empirical eigenfunctions from the information matrix, and in principle it can be substituted by many other snapshot-based model reduction techniques in [3]. Algorithm 2 lists the procedures of the RPI approach.

Algorithm 2 Reduced Picard iteration (RPI)

Require: The initial value problem (2.1).

Ensure: An approximating solution $\hat{x}(t)$.

Set a test function $\hat{x}^0(t)$ as the trial solution. Initialize the iteration number $j = 0$.

repeat

1: Update the iteration number $j = j + 1$.

2: Assemble snapshots of an approximating solution $\hat{x}^{j-1}(t)$ into matrix form \hat{X}^{j-1}

3: Compute vector field matrix \hat{F}^{j-1} whose columns associated with snapshots in \hat{X}^{j-1}

4: Form an information matrix for the extended data ensemble $\hat{Y}^{j-1} = [\hat{X}^{j-1}, \gamma \hat{F}^{j-1}]$.

5: Based on \hat{Y}^{j-1} , compute the empirical eigenfunctions Φ^j through POD or other model reduction techniques.

6: Project the original equation onto a linear subspace spanned by Φ^j and form a reduced model (3.6).

7: Use (3.7) to evolve the approximating solution $z^j(t)$ for all $t \in J^j$.

8: Express the updated solution in the original coordinate system $\hat{x}^j(t) = \Phi^j z^j(t)$.

until $\|\hat{x}^j - \hat{x}^{j-1}\|_\infty < \epsilon$, where ϵ is the tolerant error.

Obtain the final solution $\hat{x}(t) = \hat{x}^j(t)$.

When the width of all the sub-intervals $\delta_i, (i = 1, \dots, m)$ approaches zero, the subspace spanned by \hat{Y}^{j-1} converges to S^j . In this case, RPI keeps all the information of the original Picard iteration, and $\hat{x}^j \approx x^j$ is valid for each iteration. Furthermore, time integration in algorithm 2 is carried out in a reduced subspace, resulting in significant computational saving as compared with algorithm 1.

TABLE 4.1

Comparison of direct method, Picard iteration and implicit Picard iteration methods. Each technique has the full model (first row), the reduced model in the original coordinate system (second row), and the reduced model in the subspace coordinate system (third row).

direct method	Picard iteration	implicit Picard iteration
$\dot{x} = f(t, x)$ (2.1)	$\dot{x}^j = f(t, x^{j-1})$ (3.4)	$\dot{x} = f(t, x)$ (2.1)
$\hat{x} = P f(t, \hat{x})$ (2.6)	$\hat{x}^j = P^j f(t, P^j \hat{x}^{j-1})$ (3.5)	$\hat{x}^j = P^j f(t, \hat{x}^j)$ (4.1)
$\dot{z} = \Phi^T f(t, \Phi z)$ (2.5)	$\dot{z}^j = (\Phi^j)^T f(t, \Phi^j z^{j-1})$ (3.6)	$\dot{z}^j = (\Phi^j)^T f(t, \Phi^j z^j)$ (4.2)

4. Implicitly Reduced Picard Iteration (IRPI) Approach. Although RPI achieves significant saving as compared with the original Picard iteration, it still has room for substantial improvement. First of all, RPI requires allocating large memories to record the entire trajectory for each iteration in order to obtain a better approximation for the next iteration. Secondly, algorithm 2 introduces an extra step, coordinate transformation (3.8) from S^{j-1} to S^j , for each iteration which introduces extra computational overhead. For these reasons, an implicit time integration scheme is applied for a modified version of RPI, in which time integration is computed independently with the approximating trajectory obtained in the previous iteration.

4.1. Algorithm. Algorithm 2 uses an explicit time integration scheme to iteratively update the approximating solution (3.7). For the purpose of memory saving and avoiding coordinate transformation from one subspace to another, a much more efficient algorithm, implicitly reduced Picard iteration (IRPI) method, is proposed in this section. It is noticed that if \hat{x}^{j-1} is replaced by \hat{x}^j , then the reduced model (3.5) becomes

$$(4.1) \quad \hat{x}^j = P^j f(t, \hat{x}^j); \quad \hat{x}_0^j = P^j(x_0).$$

This can be considered as a modified version of RPI. However, time integration (3.7) can not be calculated explicitly here. Let $z^j = (\Phi^j)^T \hat{x}^j$, an analogy to (3.6) gives the reduced model in the subspace coordinate system for iteration j ,

$$(4.2) \quad \dot{z}^j = (\Phi^j)^T f(t, \Phi^j z^j); \quad z_0^j = (\Phi^j)^T(x_0).$$

Equations (4.1) and (4.2) can be directly obtained by projecting the original system (2.1) onto S^j . Compared with Picard iteration and RPI, IRPI only records a set of empirical eigenfunctions rather than the entire trajectory at each iteration, and seeks a better approximation by projecting the full model onto an updated subspace spanned by these empirical eigenfunctions at each iteration. Algorithm 3 reflects this change and list comprehensive procedures of the IRPI technique.

Table 4.1 compares the governing equations for these three techniques: direct method, Picard iteration and implicit Picard iteration. The full model of IRPI is the same as the one of the direct method. However, different from the reduced model of direct method, IRPI is an iterative solver, using an updated projection matrix at each iteration.

In the rest of this subsection, we discuss the geometric meaning of this algorithm. During each iteration, a new subspace S^j is constructed followed by an approximating solution $\hat{x}^{j-1}(t)$. As $\hat{x}^j(t) \rightarrow x(t)$, $S^j \rightarrow S^*$. For this reason, IRPI is an iterative

Algorithm 3 Implicitly reduced Picard iteration (IRPI)

Require: The initial value problem (2.1).

Ensure: An approximating solution $\hat{x}(t)$.

Set a test function $\hat{x}^0(t)$ as the trial solution. Initialize the iteration number $j = 0$.

repeat

1: Update the iteration number $j = j + 1$.

2: Assemble snapshots of an approximating solution $\hat{x}^{j-1}(t)$ into matrix form \hat{X}^{j-1} .

3: Compute vector field matrix \hat{F}^{j-1} associated with snapshots in \hat{X}^{j-1} .

4: Form an information matrix for the extended data ensemble $\hat{Y}^{j-1} = [\hat{X}^{j-1}, \gamma \hat{F}^{j-1}]$.

5: Based on \hat{Y}^{j-1} , compute the empirical eigenfunctions Φ^j through POD or other model reduction techniques.

6: Project the original equation onto a linear subspace spanned by Φ^j and form a reduced model (4.2).

7: Solve the reduced model and obtain an approximating solution $z^j(t)$ in the subspace coordinate system for all $t \in J^j$.

8: Express the updated solution in the original coordinate system $\hat{x}^j(t) = \Phi^j z^j(t)$.

until $\|\hat{x}^j - \hat{x}^{j-1}\|_\infty < \epsilon$, where ϵ is the tolerant error.

Obtain the final solution $\hat{x}(t) = \hat{x}^j(t)$.

manifold learning procedure, which approximates the invariant subspace by a series of subspaces. For a complete iteration cycle we begin with collection of snapshots from the previous iteration (or an initial test function), then construct a subspace spanned by a matrix which contains both the solution and the associated vector field, and then generate empirical eigenfunctions by POD or other techniques, and end with solving a reduced-order equation.

According to algorithm 3, a trial solution is given initially. One simplest approach is to set it as a constant, i.e., $\hat{x}^0(t) = x_0$. Thus, S^1 is spanned by the initial condition x_0 and the initial vector field $f(0, x_0)$. Then, the initial empirical eigenfunctions Φ^1 can be calculated by SVD of the initial information matrix $\hat{Y}^0 = [x_0, f(0, x_0)]$. Projecting the full system onto the S^1 produces a reduced model. Consequently, a better approximating solution $\hat{x}^1(t)$ can be obtained. In fact, solution vectors near the starting point approximately reside in S^1 , which can be justified by Taylor expansion of $x(t)$ to the first order approximation, $x(t) = x_0 + f(0, x_0)t + \mathcal{O}(t^2)$. However, many consequent states do not reside in S^1 and the error in $\hat{x}^1(t)$ is expected to increase significantly after a few steps. In the second iteration, the aim is to obtain a better approximation S^2 for the invariant subspace S^* . The solution matrix \hat{X}^1 can be constructed by extracting m solution vectors from the $\hat{x}^1(t)$. A large number for m will lead to intensive computation, but the selected snapshots should reflect the main dimensions of the orbit. Since all the projected solution vectors in \hat{X}^1 exactly lie on S^1 , decomposition of \hat{X}^1 through SVD will not achieve empirical eigenfunctions that can span a different subspace. Thus, we seek some other solution vectors that do not lie on S^1 . Notice that if $\hat{x}(t_i)$ ($i \in \{1, \dots, m\}$) is a solution vector that (approximately) lies on the invariant subspace S^* , so does $\hat{x}(t_i) + f(t_i, \hat{x}(t_i))\delta t$. Therefore, the matrix

$$\hat{X}^1 + \hat{F}^1 \delta t = [\dots(\hat{x}^1(t_i) + f(t_i, \hat{x}^1(t_i))\delta t)\sqrt{\delta_i} \dots],$$

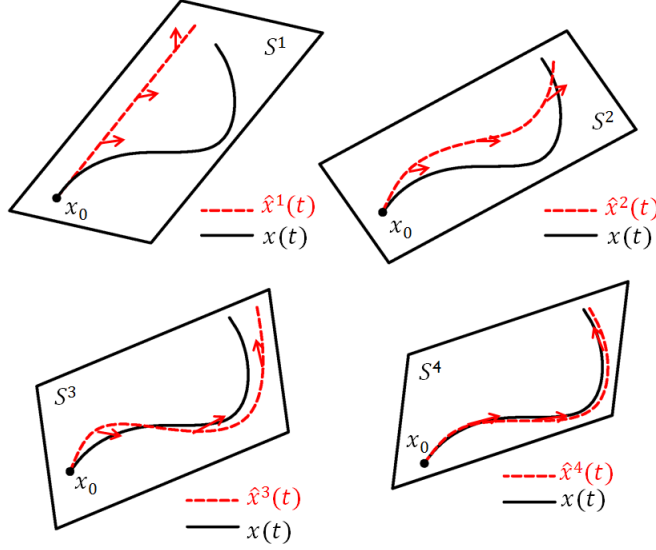


FIGURE 4.1. Graphical description of IRPI. A trial solution $\hat{x}^0(t) = x_0$ is set initially. For each iteration, a reduced model is obtained by projecting the full model onto S^j . $\hat{x}^j(t)$, represented by a curved line, is computed by the reduced model. After extracting snapshots from $\hat{x}^j(t)$ and constructing an information matrix of extended data ensemble, a better subspace S^{j+1} is obtained. A sequence of the approximating solutions $\hat{x}^j(t)$ converges to $x(t)$ asymptotically.

can also be used to provide more basis functions. Thus, a better subspace can be expressed as

$$S^2 = \text{span}(\hat{X}^1, \hat{X}^1 + \hat{F}^1 \delta t) = \text{span}(\hat{Y}^1),$$

which is the case when $j = 2$ in (3.16). S^2 is a linear subspace that contains both solution vectors obtained from iteration 1 and the neighbors of these vectors. Decomposition of \hat{Y}^1 through POD gives an updated matrix Φ^2 for empirical eigenfunctions. Then an updated solution $\hat{x}^2(t)$ can be obtained. Now, we can repeat this process to obtain solution matrix \hat{X}^2 for the next iteration. In the next subsection, we shall prove analytically that the proposed approach can achieve convergent solution for the full system.

4.2. Analytical Results. This subsection presents some analytical properties of IRPI. Since the algorithm use the classical Galerkin-POD approach for the online manifold learning, we first prove the existence and uniqueness of the solution of the reduced model formed by a Galerkin projection. Then we follow [31] to analyze the error introduced by POD. After that, we shall prove the convergence of the sequence of functions obtained by IRPI. Theorem 3.1 states that if the vector field is locally Lipschitz, then the original system exists a unique solution. This condition is also sufficient for the existence and uniqueness of the solution in the reduced model.

LEMMA 4.1 (Local Existence and Uniqueness of reduced models). *Let $B_b(x_0)$ be a closed ball of radius b around a point $x_0 \in \mathbb{R}^n$ and $f : J_0 \times B_b(x_0) \rightarrow \mathbb{R}^n$ be a uniformly Lipschitz function of x with constant K , and a continuous function of t on $J_0 = [-a, a]$. Then the associated reduced model $\dot{\hat{x}} = Pf(t, \hat{x})$ (2.6) has a unique solution at the interval $t \in J_0$ for a given initial condition $\hat{x}_0 = Px_0$, provided that*

$a = b/M$ where

$$(4.3) \quad M = \max_{(t,x) \in J_0 \times B_b(x_0)} \|f(t, x)\|.$$

Proof. In order to prove this lemma by theorem 3.1, we first show that the reduced vector field is locally Lipschitz. This is because $f(t, x)$ is a uniformly Lipschitz function for all $(t, x) \in J_0 \times B_b(x_0)$, for every point $x_1, x_2 \in B_b(x_0)$ and $t \in J_0$, then we have

$$\|f(t, x_1) - f(t, x_2)\| \leq K \|x_1 - x_2\|.$$

Recall that P is a linear projection matrix, and $\|P\| = 1$. As a consequence,

$$\|Pf(t, x_1) - Pf(t, x_2)\| \leq \|P\| \|f(t, x_1) - f(t, x_2)\| \leq K \|x_1 - x_2\|,$$

which justifies that the projected vector field $Pf(t, x)$ is Lipschitz with constant K for all $(t, x) \in J_0 \times B_b(x_0)$. Then we can apply the theorem 3.1 to yield local existence and uniqueness of the solution of the reduced-order equation $\dot{\hat{x}} = Pf(t, \hat{x})$ (2.5) for $t \in [-a', a']$, where

$$a' = \frac{b}{\max_{(t,x) \in J_0 \times B_b(x_0)} \|\Phi f(t, x)\|}.$$

Since $\|\Phi f(t, x)\| \leq \|f(t, x)\|$, we have $a' \geq b/M$ where M is defined by (4.3). Therefore, there exists a unique solution for the reduced-order equation in the interval $J = [-a, a]$, where $a = b/M$. \square

The next few theorems discuss the error of IRPI $e(t) = \hat{x}(t) - x(t)$. Since the reduced model in IRPI is formed by Galerkin projection, as discussed in Section, the $e(t)$ can be decomposed into two parts: $e_o(t)$, the component orthogonal to the projected subspace S , and $e_i(t)$, the component parallel to S . It will be shown that $e_i(t)$ depends on $e_o(t)$. Substituting (2.1) and (2.6) into the differentiation of $e_o(t) + e_i(t) = \hat{x}(t) - x(t)$ yields

$$(4.4) \quad \dot{e}_o + \dot{e}_i = Pf(t, \hat{x}) - f(t, x).$$

Using $P^2 = P$ and left multiplying (4.4) by P , we can derive the governing equation for $e_i(t)$,

$$(4.5) \quad \dot{e}_i(t) = P(f(t, x + e_o + e_i) - f(t, x)),$$

where $x(t)$ and $e_o(t)$ can be considered as forcing terms. Proposition 4.2 in [31] reveals that the error of $e_i(t)$ is bounded by $e_o(t)$ and $x(t)$. In order to introduce the following convergence theorem, we restate this property with minor modifications.

LEMMA 4.2. *Consider solving the initial value problem given by (2.1) using the reduced model in the interval $J_0 = [0, a]$. Let $P \in \mathbb{R}^{n \times n}$ be the projection matrix associated with a linear subspace S . Let $x(t)$ be the solution of the full model, $\tilde{x}(t) = x(t) + e_o(t)$ be the projection of $x(t)$ onto S , and $\hat{x}(t) = x(t) + e_o(t) + e_i(t)$ be the solution of the reduced model. Let K be the Lipschitz constant of $f(t, x)$ and*

$$\|f(t, x_1) - f(t, x_2)\| \leq K \|x_1 - x_2\|$$

for all $(t, x_1), (t, x_2) \in D \subset J_0 \times \mathbb{R}^n$, where the region D contains $(t, x(t))$, $(t, \tilde{x}(t))$ and $(t, \hat{x}(t))$ for all $t \in J_0$. Then the error $e = e_o + e_i$ in the infinity norm is bounded by

$$(4.6) \quad \|e\|_\infty \leq e^{Ka} \|e_o\|_\infty + \|e_i(0)\|.$$

Proof. The governing equation of $e_i(t)$, given by (4.5), yields

$$\begin{aligned} \|e_i(t+h)\| &= \|e_i(t) + hPf(t, x + e_o + e_i) - hPf(t, x)\| + \mathcal{O}(h^2) \\ &\leq \|e_i(t)\| + h \|Pf(t, x + e_o + e_i) - Pf(t, x + e_o)\| \\ &\quad + h \|Pf(t, x + e_o) - Pf(t, x)\| + \mathcal{O}(h^2). \end{aligned}$$

Applying the Lipschitz conditions, we get

$$\frac{\|e_i(t+h)\| - \|e_i(t)\|}{h} \leq K \|e_i(t)\| + K \|e_o(t)\| + \mathcal{O}(h).$$

Since $\mathcal{O}(h)$ can be uniformly bounded independent of $e(i)$, we can obtain

$$\|e_i(t)\| \leq K \int_0^t e^{K(t-\tau)} \|e_o(\tau)\| d\tau + \|e_i(0)\|.$$

By definition, $\|e_o\|_\infty \geq \|e_o(t)\|$ for any $t \in J_0$. It follows that

$$\begin{aligned} \|e_i\|_\infty &\leq K \|e_o\|_\infty \int_0^{0+a} e^{K(0+a-\tau)} d\tau + \|e_i(0)\| \\ &= (e^{Ka} - 1) \|e_o\|_\infty + \|e_i(0)\|. \end{aligned}$$

Using $e(t) = e_o(t) + e_i(t)$, we have

$$\|e\|_\infty \leq \|e_i\|_\infty + \|e_o\|_\infty \leq e^{Ka} \|e_o\|_\infty + \|e_i(0)\|.$$

Therefore, an upper bound for $\|e\|_\infty$ is obtained. \square

Lemma 4.2 implies that as long as $e_o^j(t) \rightarrow 0$ as $j \rightarrow \infty$, then $e^j(t) \rightarrow 0$. Now, we are ready to discuss the main property of IRPI.

THEOREM 4.3 (Local Convergence of IRPI). *Consider solving the initial value problem given by (2.1). Let $B_b(x_0)$ denote a closed ball of radius b around a point $x_0 \in \mathbb{R}^n$ and J_0 be a closed time interval, $J_0 = [0, a]$. Suppose $f(t, x)$ is a uniformly Lipschitz function with constant K for all $(t, x) \in J_0 \times B_b(x_0)$. The IRPI approach is applied to obtain an approximating solution. At iteration j , let $x(t)$ be the solution of the full model, $\tilde{x}^j(t) = x(t) + e_o^j(t)$ be the projection of $x(t)$ onto S^j , and $\hat{x}^j(t) = x(t) + e_o^j(t) + e_i^j(t)$ be the solution of the reduced model. In an ideal situation, $x_0 \in S^j$, while the solution and vector field satisfy $\hat{x}^{j-1}(t) \subset S^j$ and $f(t, \hat{x}^{j-1}) \subset S^j$ for all $t \in J_0$, respectively. Then the sequence of functions $\hat{x}^j(t)$ uniformly converges to $x(t)$ for all $t \in J_0$ provided that*

$$(4.7) \quad a < \min \left[\frac{b}{M}, \frac{e^{-Kb/M}}{K} \right],$$

where M is given by (4.3).

Proof. Since $f(t, x)$ is a uniformly Lipschitz function for all $(t, x) \in J_0 \times B_b(x_0)$ with constant K , theorem 3.1 implies the existence and uniqueness of the solution $x(t)$ of the full model (2.1). Then we can uniquely determine $\tilde{x}(t)$ by $\tilde{x}(t) = P^j x(t)$. Meanwhile, by lemma 4.1, the projected vector field $Pf(t, x)$ is also Lipschitz with constant K for all $(t, x) \in J_0 \times B_b(x_0)$ and there exists a unique solution $\hat{x}^j(t)$ for the reduced model (3.6). Therefore, $x(t)$, $\tilde{x}(t)$ and $\hat{x}^j(t)$ uniquely exist for all $t \in J_0$.

Next we discuss the error $e(t) = \hat{x}^j(t) - x(t)$ of the reduced model in the interval J_0 . As mentioned before, the error has orthogonal components $e_o(t)$ and parallel components $e_i(t)$. Multiplying (4.4) on the left by $I - P^j$, we obtain the governing equation for $e_o(t)$,

$$(4.8) \quad \dot{e}_o^j = -(I - P^j)f(t, x).$$

Since $x_0 \in S^j$, we have $P^j x_0 = x_0$. Therefore, the initial condition is given by $e_o^j(0) = 0$. Since the vector field $f(t, \hat{x}^{j-1})$ is invariant under the projection operator P^j (see 3.10), we have

$$(I - P^j)f(t, \hat{x}^{j-1}) = 0.$$

It follows that

$$(4.9) \quad \dot{e}_o^j = (I - P^j)[f(t, x + e^{j-1}) - f(t, x)].$$

For $h > 0$, Taylor expansion of $\dot{e}_o^j(t)$ gives

$$(4.10) \quad \|e_o^j(t + h)\| \leq \|e_o^j(t)\| + \|h(I - P^j)[f(t, x + e^{j-1}) - f(t, x)]\| + \mathcal{O}(h^2).$$

Considering $x(t) \in B_b(x_0)$, $\hat{x}^{j-1}(t) \in B_b(x_0)$, and $f(t, x)$ is a uniformly Lipschitz function for all $(t, x) \in J_0 \times B_b(x_0)$ with constant K , it follows that

$$(4.11) \quad \|(I - P^j)[f(t, x + e^{j-1}) - f(t, x)]\| \leq \|f(t, x + e^{j-1}) - f(t, x)\| \leq K \|e^{j-1}(t)\|.$$

We can combine (4.10) with (4.11) to obtain

$$(4.12) \quad \frac{\|e_o^j(t + h)\| - \|e_o^j(t)\|}{h} \leq K \|e^{j-1}(t)\| + \mathcal{O}(h),$$

where the $\mathcal{O}(h)$ term may be uniformly bounded independent of $e_o^{j-1}(t)$. Integrating (4.12) with respect to t yields

$$(4.13) \quad \|e_o^j(t)\| \leq K \int_0^t \|e^{j-1}(\tau)\| \, d\tau \leq Ka \|e^{j-1}\|_\infty.$$

As $f(t, x)$ is a Lipschitz function for all $(t, x) \in J_0 \times B_b(x_0)$, we have $x(t) \in B_b(x_0)$, $\tilde{x}(t) \in B_b(x_0)$, and $\hat{x}(t) \in B_b(x_0)$. Notice that $e_i(0) = 0$ since the starting point \hat{x}_0^j is the projection of x_0 onto S^j .

In order to apply lemma 4.2, we will show that $x(t) \in B_b(x_0)$. Provided $a < b/M$, this is valid since

$$(4.14) \quad \|x(t) - x_0\| = \left\| \int_0^t f(\tau, x) d\tau \right\| \leq \int_0^t \|f(\tau, x)\| \, d\tau \leq Ma < b.$$

In addition, we have

$$(4.15) \quad \|\tilde{x}^j(t) - x_0\| = \|P^j x(t) - P^j x_0\| \leq \|x(t) - x_0\| < b,$$

which means that $\hat{x}^j(t) \in B_b(x_0)$. Similarly, since

$$(4.16) \quad \|\hat{x}^j(t) - x_0\| = \|\hat{x}^j(t) - P^j x_0\| = \left\| \int_0^t P f(\tau, \hat{x}^j) d\tau \right\| < b,$$

we achieve $\hat{x}^j(t) \in B_b(x_0)$ for all $t \in J_0$. As a consequence, we can combine (4.6) in lemma 4.2 with (4.13) to obtain

$$(4.17) \quad \|e^j\|_\infty \leq K a e^{K a} \|e^{j-1}\|_\infty.$$

Since

$$K a e^{K a} < K a e^{K b/M} < 1,$$

then a sequence of functions $\|e^j(t)\|$ uniformly converges to 0, which means the sequence of functions $\hat{x}^j(t)$ uniformly converges to the fixed point $x(t)$ for all $t \in J_0$. \square

REMARK 4.4. Ideally a linear subspace S^j at iteration j contains an approximating solution $x^{j-1}(t)$ with the associated vector field $f^{j-1}(t)$ of the previous iteration for all $t \in J_0$, which is the case for equations (3.9) and (3.10). However, it is more often to form S^j by extracting the first few dominant modes from the matrix of extended data ensemble (3.16). In this circumstance, (3.9) and (3.10) should be replaced by

$$(4.18) \quad P^j(\hat{x}^{j-1}) \approx \hat{x}^{j-1},$$

and

$$(4.19) \quad P^j(f(t, \hat{x}^{j-1})) \approx f(t, \hat{x}^{j-1}),$$

respectively. To quantify the error from this modification, we define

$$(4.20) \quad \varepsilon := \frac{1}{a} \int_0^a \|(I - P^j) \hat{x}^{j-1}(\tau)\|^2 d\tau + \frac{\gamma^2}{a} \int_0^a \|(I - P^j) f(\tau, \hat{x}^{j-1})\|^2 d\tau.$$

If SVD is used to construct the POD basis, ε can be estimated by

$$(4.21) \quad \varepsilon = \sum_{i=k_j+1}^n \lambda_i^2,$$

where k^j is the dimension of subspace S^j and λ_i^2 is the i th eigenvalue of autocorrelation matrix for the extended data ensemble

$$(4.22) \quad Y^j = [x^{j-1}(t), \gamma f(t, x^{j-1})].$$

If the fraction error in energy truncation that SVD gives is bounded by $1 - \eta$, one obtains

$$(4.23) \quad \varepsilon < (1 - \eta) E_r^j,$$

where E_r^j , as defined in (2.18), denotes the total energy of the extended data ensemble Y^j . In practice, it is more tractable to take the discrete form \hat{Y}^j (3.16) as an approximation for Y^j . Accordingly, a more strict bound on ε is given by

$$(4.24) \quad \varepsilon < \rho(1 - \eta) E_r^j,$$

where ρ is a coefficient associated with temporal discretization. Let δ be the maximal time interval among the neighboring snapshots. If $f(t, x) \in C^1(J_0 \times B_b(x_0))$, then we can obtain

$$(4.25) \quad \rho = 1 + \frac{\delta}{a} \max_{t \in J_0} \left(\frac{\|\dot{f}(t, \hat{x}^{j-1})\|}{\|f(t, \hat{x}^{j-1})\|}, \frac{\|f(t, \hat{x}^{j-1})\|}{\|\hat{x}^{j-1}(t)\|} \right) + \mathcal{O}\left(\frac{\delta^2}{a^2}\right),$$

which implies that $\rho \rightarrow 1$ as $\delta \rightarrow 0$. Since the IRPI approach uses a snapshots-based reduced model, such as POD, for each iteration, consequently the sequence of functions can not converge to the exact solution of the original system. However, as proposition 4.5 indicate, we can always bound the error of the approximating solution obtained by IRPI.

PROPOSITION 4.5. *Consider solving the initial value problem given by (2.1). Let $B_b(x_0)$ denote a closed ball of radius b around a point $x_0 \in \mathbb{R}^n$ and J_0 be a closed time interval, $J_0 = [0, a]$. Suppose $f(t, x)$ is a uniformly Lipschitz function with constant K for all $(t, x) \in J_0 \times B_b(x_0)$. The IRPI approach is applied to obtain an approximating solution. For iteration j , empirical eigenfunctions, which is used to span a linear subspace S^j , are calculated by SVD of the information matrix \hat{Y}^j (3.16). Let $x(t)$ be the solution of the full model, $\tilde{x}^j(t) = x(t) + e_o^j(t)$ be the projection of $x(t)$ onto S^j , and $\hat{x}^j(t) = x(t) + e_o^j(t) + e_i^j(t)$ be the solution of the reduced model. Then the sequence of functions $\hat{x}^j(t)$ defined by IRPI approaches to $x(t)$ with an upper bound for $\|e\|_\infty$ given by*

$$(4.26) \quad \chi = \frac{a\epsilon e^{Ka}}{\gamma^2(1 - Kae^{Ka})} + \frac{(1 + e^{Ka})\theta}{1 - Kae^{Ka}},$$

for all $t \in J_0$ provided that

$$(4.27) \quad a < \min \left[\frac{b}{2M}, \frac{e^{-Kb/2M}}{K} \right],$$

where M is given by (4.3), θ is the maximal error of the initial condition in the reduced model, and $\theta < b/2$.

Proof. Here we follow a similar approach to the proof of theorem 4.3. As proved in theorem 4.3, $x(t)$ and $\tilde{x}(t)$ uniquely exist in the interval J_0 . Moreover, $x(t) \in B_b(x_0)$ and $\tilde{x}^{j-1}(t) \in B_b(x_0)$. In a general situation, the initial condition of the reduced model (3.5) can be expressed as $\hat{x}_0^j = x_0 + e^j(0)$. $\hat{x}^j(t)$, the solution of nearby initial condition, still uniquely exists. Moreover, $\hat{x}^{j-1}(t) \in B_b(x_0)$, which can be justified by

$$(4.28) \quad \|\hat{x}^j(t) - \hat{x}_0^j\| \leq \|\hat{x}^j(t) - \hat{x}_0^j\| + \|\hat{x}_0^j - x_0\| < b/2 + b/2 = b.$$

Compare with (4.8), the governing equation of $e_o(t)$ has a more complex form,

$$\dot{e}_o^j = (I - P^j)[f(t, x + e^{j-1}) - f(t, x)] - (I - P^j)f(t, x + e^{j-1}).$$

Under a non-ideal situation, the second term on the right hand side is nonzero since (3.10) is no longer valid. For the same reason, the initial value of $e_o^j(0)$ is nonzero. Expanding $\dot{e}_o^j(t)$ and using the inequality (4.11), one obtain

$$(4.29) \quad \frac{\|e_o^j(t+h) - e_o^j(t)\|}{h} \leq K \|e^{j-1}(t)\| + \|(I - P^j)f(t, x + e^{j-1})\| + \mathcal{O}(h),$$

where the $\mathcal{O}(h)$ term may be uniformly bounded independent of $e_o^{j-1}(t)$. Integrating (4.29) with respect to t yields

$$(4.30) \quad \|e_o^j(t)\| \leq K \int_0^t \|e^{j-1}(\tau)\| \, d\tau + \int_0^t \|(I - P^j)f(\tau, x + e^{j-1})\| \, d\tau + \|e_o^j(0)\|.$$

As shown in (4.13), the first term on the right hand side is bounded by $Ka\|e^{j-1}\|_\infty$. By the definition of ε (4.20), the second term on the right hand side is also bounded for all $t \in J_0$,

$$(4.31) \quad \int_0^t \|(I - P^j)f(\tau, x + e^{j-1})\| \, d\tau < a\varepsilon/\gamma^2.$$

Combining (4.6), (4.13), (4.30) and (4.31), one obtains

$$(4.32) \quad \|e^j\|_\infty < Ka e^{Ka} \|e^{j-1}\|_\infty + \frac{a\varepsilon e^{Ka}}{\gamma^2} + e^{Ka} \|e_o^j(0)\| + \|e_i^j(0)\|.$$

If the initial error is bounded by

$$\|e_o^j(0)\| \leq \theta_o, \quad \|e_i^j(0)\| \leq \theta_i,$$

then

$$(4.33) \quad \|e^j\|_\infty < Ka e^{Ka} \|e^{j-1}\|_\infty + \frac{a\varepsilon e^{Ka}}{\gamma^2} + \theta_o e^{Ka} + \theta_i.$$

Considering

$$Ka e^{Ka} < Ka e^{Kb/2M} < 1,$$

the total error is bounded by

$$(4.34) \quad \|e^j\|_\infty < \frac{a\varepsilon e^{Ka}}{\gamma^2(1 - Ka e^{Ka})} + \frac{\theta_o e^{Ka}}{1 - Ka e^{Ka}} + \frac{\theta_i}{1 - Ka e^{Ka}}.$$

Since $\theta \geq \theta_o$ and $\theta \geq \theta_i$, then a sequence of functions $\|e^j(t)\|$ is bounded by a constant χ as $j \rightarrow \infty$, where χ is given by (4.26). \square

REMARK 4.6. Proposition 4.5 provides an local error bound for the IRPI method. The first term on the right hand side of (4.34) is introduced by the truncation error. By decreasing the width of time intervals among neighboring snapshots and increasing the number of POD modes, we can limit the value of ε smaller than any positive real numbers. The second term is the magnified error caused by $e_o^j(0)$, the initial error orthogonal to S^j . If x_0 does not resides in S^j , then $e_o^j(0) = (I - P^j)x_0 \neq 0$. The third term in (4.34) comes from the initial error on S^j . It vanishes if the projection of x_0 onto S^j is the starting point for the reduced model, i.e., $\hat{x}_0^j = Px_0$. However, if \hat{x}_0^j is obtained by numerical calculation from a previous time, the third term can be nonzero. In order to balance the truncation error of the vector field and initial condition, γ in (3.16) can be set to $\gamma = 1$. When $\varepsilon = 0$, and $\|e_o^j(0)\| = 0$, proposition 4.5 degenerates to the case in theorem 4.3, i.e. $\chi = 0$, which means the sequence of functions $\hat{x}^j(t)$ converges to the fixed point $x(t)$ for all $t \in J_0$.

At the first glance, the error bound χ is only determined by the initial error θ and truncation error of the vector field. The proof of convergence in theorem 4.3 and its proposition 4.5 do not require that $\hat{x}^{j-1}(t)$ resides in S^j for any $t \in (0, T]$. As an alternative to (3.16), a more straightforward form of the information matrix for the extended data ensemble can be written as

$$(4.35) \quad \tilde{Y}^j := [x_0, \gamma \hat{F}^j].$$

In this case, the new sequence of functions can still converge to $x(t)$. The main reason we use (3.16) is that we want to find an optimal subspace that minimize the error of the reduced model for the entire trajectory. Specifically, as a Picard-type iteration, IRPI can only guarantee that the next iteration can decrease the error for a limit time interval $J_0 = [0, a]$. Then, (4.17) can be rewritten as

$$(4.36) \quad \frac{\|e^j\|_\infty}{\|e^{j-1}\|_\infty} \leq Kae^{Ka}.$$

When $a > 1/K$, the left side might be greater than 1. Thus, although $\hat{x}^j(x)$ has less error locally, it might be worse than $\hat{x}^{j-1}(x)$ in a global sense. By adding consequent solution vectors in the extended data ensemble, we have $\hat{x}^{j-1} \subset S^j$. Consequently,

$$(4.37) \quad \|e_o^j(t)\| \leq \|e^{j-1}(t)\|.$$

Using (4.6) in lemma 4.2, and assume $\|e_i(0)\| = 0$, one obtains

$$(4.38) \quad \frac{\|e^j\|_\infty}{\|e^{j-1}\|_\infty} \leq e^{Ka}.$$

(4.38) still can not grantee that the $\|e^{j-1}(t)\| < \|e^j(t)\|$ for all $t \in J$. However, (4.38) provide a stronger bound than (4.36) when $t > 1/K$, which can effectively decrease the global error.

Sofar, we proved the convergence of IRPI for a local time interval $J_0 = [0, a]$. Since the estimates used to obtain J_0 are certainly not optimal, the true convergence time interval is usually much larger. Suppose $J = [0, T] \subset J_m$, where J_m is the maximal interval of existence of the original solution. We now prove that the convergence region J_0 can be extended to J .

THEOREM 4.7 (Global Convergence of IRPI). *Consider solving the initial value problem given by (2.1) for the interval $J = [0, T]$. Suppose the vector field $f(t, x)$ is locally Lipschitz on D' , where D' is an open set that contains $(t, x(t))$ for all $t \in J$. The IRPI approach is applied to obtain an approximating solution. At iteration j , let $x(t)$ be the solution of the full model, $\hat{x}^j(t) = x(t) + e_o^j(t)$ be the projection of $x(t)$ onto S^j , and $\hat{x}^j(t) = x(t) + e_o^j(t) + e_i^j(t)$ be the solution of the reduced model. In an ideal situation, $x_0 \in S^j$. Furthermore, the solution and vector field satisfy $\hat{x}^{j-1}(t) \subset S^j$ and $f(t, \hat{x}^{j-1}) \subset S^j$ for all $t \in J^{j-1}$, respectively. Then the sequence of functions $\hat{x}^j(t)$ uniformly converges to $x(t)$ for all $t \in J$.*

Proof. We first define a closed set $E = \{(t, x(t)) : t \in J\} \subset D'$. Let d be the distance between E and the boundary of D' . Since D' is an open set, we have $d > 0$. Thus, we can choose a positive constant b such that $0 < b < d$. Since $f(t, x)$ is locally Lipschitz on D' and $E' = \{(t, y) : t \in J, y \in B_b(x(t))\} \subset D'$ is compact, then $f(t, x)$ is Lipschitz on E' . Let K denote the Lipschitz constant for $(t, x) \in E'$. In

addition, we can choose the value of a bounded by (4.7). Suppose $J^* \subset [0, T]$ be the maximal interval containing $t = 0$ such that for all $t \in J^*$, $\hat{x}^j(t) \rightarrow x(t)$ uniformly as $j \rightarrow \infty$. Theorem 4.3 indicates that IRPI will generate a sequence of functions $\hat{x}^j(t)$ that uniformly converges to $x(t)$ for all $t \in J_0 = [0, a]$. For this reason, we have $J^* \neq \emptyset$.

Now assume $J^* \neq J$. Then there exists a $t_i \in J$ such that $t_i \in J^*$, $t_i + a \leq T$ and $t_i + a \notin J^*$. $t_i \in J^*$ means for every $\kappa > 0$ there exists a integer $M_1(\kappa) > 0$ such that for all j with $j > M_1(\kappa)$, $\hat{x}^j(t_i)$ uniquely exists and $\|\hat{x}^j(t_i) - x(t_i)\| < \kappa$.

Considering the initial value problem

$$(4.39) \quad \dot{y} = f(t, y); \quad y(0) = y_0 = x(t_i).$$

The corresponding reduced model of IRPI at iteration l is given by

$$(4.40) \quad \dot{\hat{y}}^l = P^l f(t, \hat{y}^l); \quad \hat{y}^l(0) = y_0 + e^l,$$

and $\|e^l\| < \kappa'$. Proposition 4.5 implies that for all $t \in J_0 = [0, a]$, $\|\hat{y}^l(t) - y(t)\| < \chi$ as $l \rightarrow \infty$, where χ is given by 4.26. Formally, for every $\kappa' > 0$, there is another positive integer $M_2(\kappa')$ such that whenever $l > M_2(\kappa')$, then $\|\hat{y}^l(t) - y(t)\| < \chi + \kappa'$. Plug (4.26) into this inequity, we have

$$(4.41) \quad \|\hat{y}^l(t) - y(t)\| \leq \frac{a\varepsilon e^{Ka}}{\gamma^2(1 - Kae^{Ka})} + \frac{(1 + e^{Ka})\kappa}{1 - Kae^{Ka}} + \kappa',$$

for any $t \in J_0$. In an ideal case, the truncation error is zero, i.e., $\varepsilon = 0$. Then, the right hand side of (4.41) can be arbitrarily small. Note that $y(t) = x(t_i + t)$ and $\hat{y}^j(t) = \hat{x}^j(t_i + t)$. Therefore, for every $\epsilon > 0$, there exist an integer

$$(4.42) \quad N(\epsilon) = M_1 \left(\frac{(1 - Kae^{Ka})\epsilon}{2 + 2e^{Ka}} \right) + M_2 \left(\frac{\epsilon}{2} \right),$$

as long as $j > N(\epsilon)$, then $\|\hat{x}^j(t) - x(t)\| \leq \epsilon$ for all $t \in [0, t_i + a]$. However, this contradicts with our assumption that $t_i + a < T$ and $t_i + a \notin J^*$. Therefore, $J^* = J$, i.e., $\hat{x}^j(t)$ uniformly converges to $x^t(t)$ for all $t \in J$. \square

4.3. Computational Cost. In the previous subsection, we have shown that IRPI results in a series of approximating solutions that converges to an actual solution. Next, we will study the computational cost associated with the proposed IRPI method. In fact, a reduced-order equation formed by Galerkin projection can obtain fast computation only when the analytical formula of reduced vector field $\Phi^T f(t, \Phi z)$ can be simplified, especially when $f(t, x)$ is a polynomial in x . Furthermore, since IRPI is an iterative solver, IRPI can not achieve speedups unless it obtains convergent results in a few iterations. This subsection analyzes the cost of IRPI for solving an initial value problem, then gives some clues for optimized applications.

Assume an initial trial solution is given. In algorithm 3, the cost of IRPI for each iteration mainly involves four procedures: construct of the information matrix for an extended data ensemble (C_1 , step 2-4), compute of empirical eigenfunctions (C_2 , step 5), project the original system onto a subspace and solve the reduced model (C_3 , step 6-7), obtain an approximating solution in the original coordinate system (C_4 , step 8). The total cost is

$$(4.43) \quad C = R(C_1 + C_2 + C_3 + C_4),$$

where R is the number of iterations.

In terms of computational cost, C_3 takes the main contribution. Let $\gamma(n)$ be the cost involved in computing $f(t, x)$, the vector field in \mathbb{R}^n . The cost of computing vector field in a reduced subspace, $\Phi^T f(t, \Phi z)$, is denoted by $\hat{\gamma}(k, n)$. It is possible that $\hat{\gamma}(k, n) \geq \gamma(n)$ if $f(t, x)$ is not a polynomial operator. If an explicit scheme is applied for time integration,

$$(4.44) \quad C_3 = \mathcal{O}(N\hat{\gamma}(k, n)),$$

where N is the total number of time steps.

The other procedures are carried out only once at the beginning of each iteration. Since an extended data ensemble is computed in the original space, the computational cost to compute the vector field at m snapshots is

$$(4.45) \quad C_1 = \mathcal{O}(m\gamma(n)).$$

$m \ll T$ should be satisfied for cost saving purposes. The cost of constructing a set of empirical eigenfunctions depends on the specified technique used. Suppose $m \ll n$, POD basis can be obtained by $\mathcal{O}(h^2n)$ flops by SVD [41], where h denote the number of columns in the information matrix. Since the information matrix defined by IRPI has $2m$ columns, we have

$$(4.46) \quad C_2 = \mathcal{O}(4m^2n).$$

Finally, in order to obtain an approximating solution in the original coordinate system and collect the snapshots for the next iteration, we need compute $\hat{x}(t_i) = \Phi z(t_i)$ for m snapshots. The corresponding cost is

$$(4.47) \quad C_4 = \mathcal{O}(mnk).$$

Inserting equations (4.45)-(4.47) into (4.48), one can obtain the total cost for IRPI.

$$(4.48) \quad C = \mathcal{O}(R \cdot (m\gamma(n) + 4m^2n + N\hat{\gamma}(k, n) + mnk)).$$

C_2 and C_3 are inherited from the classical Galerkin-POD approach while construction of the extended data ensemble need an extra cost C_1 and C_4 . Usually, $C_1 + C_4 \ll C_2 + C_3$. Since algorithm 3 is an iterative algorithm, it introduce a fact R for iteration times. Considering the cost of the full model is given by $\mathcal{O}(N\gamma(n))$ for an explicit time integration, roughly speaking if $R \cdot C_3 \ll \mathcal{O}(N\gamma(n))$, IRPI can obtain significantly speedup.

Sometimes an implicit scheme for integrating an ODE is preferable to an explicit method for stability reasons, and also due to capability of using larger time steps during time integration. In terms of computation cost, one only need update the expression for C_3 . More discussion of the cost involves Galerkin-POD approach appears in [31]. Kevrekidis

It has been noticed that even for a polynomial vector field, POD-Galerkin approach might not be able to significantly accelerate computation since it results in small but full matrices (tensors) while common discretization techniques usually generate large but sparse ones. This effect is even considerably increased when treating with POD based IRPI, whose computational cost is roughly R times the standard POD-Galerkin approach. For the purpose of computational saving, it is preferable

to substitute standard POD by a more efficient snapshot-based technique for dimensional reduction in algorithm 3. Within the framework of IRPI, one simple approach to effectively decrease iteration number R is to set up a “good” trial solution initially. For example, if the full model is obtained by finite difference method with a large number of grid points, a coarse model can be set as the same scheme with smaller number of grid points and a larger time step. Instead of setting up the initial trial solution to be a constant, a better trial solution can be obtained by the coarse model. This idea is illustrated in simulations in the next subsection.

4.4. Numerical Results. The proposed algorithm, IRPI, is illustrated in this subsection by two numerical examples: linear advection-diffusion equation and non-linear Burgers equation. Both examples focus on demonstrating the capability of IRPI to deliver accurate results using reduced models. We also show an application of IRPI for posteriori error estimation of a coarse model or reduced model.

Let $u = u(t, x)$ denote a scalar field. Consider the advection-diffusion equation with constant moving speed c and diffusion coefficient ν ,

$$(4.49) \quad \frac{\partial}{\partial t} u = -c \frac{\partial}{\partial x} u + \nu \frac{\partial^2}{\partial x^2} u,$$

on space $x \in [0, 1]$. Without loss of generality, periodic boundary condition is applied,

$$(4.50) \quad \begin{aligned} u(t, 0) &= u(t, 1), \\ u_x(t, 0) &= u_x(t, 1). \end{aligned}$$

The initial condition is provided by a Gaussian function

$$(4.51) \quad u(0, x) = e^{-\frac{1}{2} \left(\frac{x-1/3}{1/5} \right)^2}.$$

The fully resolved model in higher dimensional space is obtained through a high resolution finite difference simulation with spatial discretization by n equally spaced grid points. The advection term is discretized by an explicit first-order upwind scheme while the diffusion term is calculated using the implicit Crank-Nicolson scheme with second-order central difference. This results in an ODE of the form,

$$(4.52) \quad \dot{y} = Ay.$$

We slightly abuse the notation and denote $y \in \mathbb{R}^n$ as the discretized state vector, $A \in \mathbb{R}^{n \times n}$ is a linear operator in the original space. The associated reduced-order equation is

$$(4.53) \quad \dot{z} = A'z$$

where $z \in \mathbb{R}^k$ is the representation of state in a reduced space, $A' \in \mathbb{R}^{k \times k}$ is the reduced operator and $k \ll n$.

For our numerical experiments, we consider a system with $c = 0.5$ and $\nu = 10^{-3}$, which gives rise to a system with diffusion as well as advection propagating to the right. This can be seen from Figure 4.2(a), where the initial condition as well as the solution at $t = 1$ are shown. The full model (reference benchmark solver) is computed through $n = 500$ grid points. In order to initialize IRPI, a smaller simulation is carried out by finite difference method with a coarse grid of $k = 20$ with larger time step. A filtered solution is constructed by extracting the first 10 Fourier modes, which is

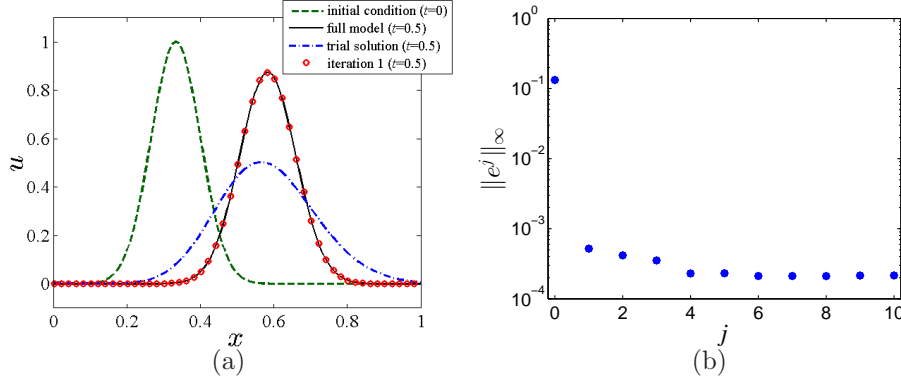


FIGURE 4.2. (Color online.) (a) The velocity profiles at $t = 0$ and $t = 0.5$ of the one-dimensional advection-diffusion equation with constant speed $c = 0.5$ and diffusion coefficient $\nu = 10^{-3}$. $N = 500$ grid points are used to obtain the full model for the fixed space domain $[0, 1]$. The coarse solution is obtained by extracting the first 10 Fourier modes from the coarse simulation based on $n = 20$ grid points. (b) Convergence of IRPI. Plot of the maximal difference between the benchmark solution $u(t)$ the iterative solution $\hat{u}^j(t)$ for $t \in [0, 0.5]$, $\|e^j\|_\infty = \sup\{\|\hat{u}^j(t) - u(t)\|_\infty : t \in [0, 0.5]\}$. Roughly speaking It takes only 1 iteration for IRPI to obtain the convergent solution.

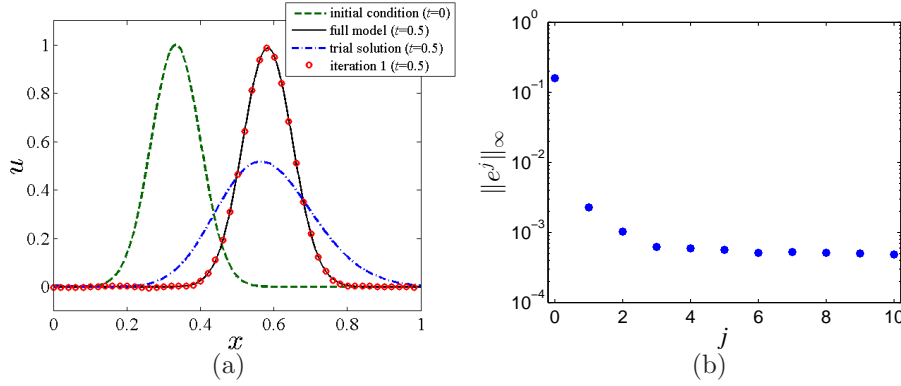


FIGURE 4.3. (Color online.) (a) The velocity profiles at $t = 0$ and $t = 0.5$ of the one-dimensional advection equation with constant speed $c = 0.5$. $N = 1000$ grid points are used to obtain the full model for the fixed space domain $[0, 1]$. The coarse solution is obtained by extracting the first 10 Fourier modes from the coarse simulation based on $n = 20$ grid points. (b) Convergence of IRPI. Plot of the maximal difference between the benchmark solution $u(t)$ and the iterative solution $\hat{u}^j(t)$, $\|e^j\|_\infty = \sup\{\|\hat{u}^j(t) - u(t)\|_\infty : t \in [0, 0.5]\}$. Roughly speaking It takes only 1 iteration for IRPI to obtain the convergent solution.

used as the initially trial solution for the IRPI method. During the first iteration, the full-order equation is projected onto a subspace spanned by $k = 13$ dominant modes and a better approximation is obtained. The convergence plot for IRPI is shown in Figure 4.2(b), which accords with our theoretical analysis that the sequence of the functions $\hat{u}^j(t)$ approaches to the actual solution $u(t)$ after a few iterations.

When $\nu \rightarrow 0$ the advection-diffusion equation will degenerate to the advection equation. We considered the system with $c = 0.5$ and $\nu = 0$. The initial Gaussian wave propagates to the right without any dissipations. This can be seen from Figure 4.3(a), where the initial condition as well as the solution at $t = 1$ are shown. The full model (reference benchmark solver) is computed through $n = 1000$ grid points. We still use $k = 20$ grid points for a coarse simulation and take the first 10 Fourier

modes to construct the trial solution. Different from the actual solution, the trial solution has significant diffusion effect. During the first iteration, the full-order equation is projected onto a subspace spanned by $k = 12$ dominant modes and a better approximation is obtained. Figure 4.3(b) compares the maximal difference between the benchmark solution $u(t)$ the iterative solution $\hat{u}^j(t)$ for $t \in [0, 0.5]$ in the first 10 iterations.

Viscous Burgers equation is very similar to the advection-diffusion equation except that the advection velocity is no longer constant. The general form of the one-dimensional Burgers equation is given as

$$(4.54) \quad \frac{\partial}{\partial t} u = -u \frac{\partial}{\partial x} u + \nu \frac{\partial^2}{\partial x^2} u,$$

where ν is the diffusion coefficient. Let $\Omega = [0, 1]$ denotes the computational domain. Periodic boundary conditions (4.50) are applied. The Gaussian function (4.51) is used for the initial condition.

The fully resolved model is obtained through a high resolution finite difference simulation with spatial discretization by n equally spaced grid points. We still use the explicit first-order upwind scheme to compute the advection term and implicit Crank-Nicolson scheme with second-order central difference to compute the diffusion term. Truncating the projection of velocity field $u(t, x)$ on the linear subspace spanned by $\Phi = [\phi_1, \dots, \phi_k]$, we obtain

$$(4.55) \quad \hat{u}(t, x) = \sum_{j=1}^k z_j(t) \phi_j(x).$$

Substituting this approximation into (4.54) and using Einstein summation convention, the evolution of the reduced coordinates $z \in \mathbb{R}^k$ is given by

$$(4.56) \quad \dot{z}_l = -A_{lij} z_i z_j - B_{lj} z_j.$$

where $A_{lij} \in \mathbb{R}^{k \times k \times k}$ and $B_{lj} \in \mathbb{R}^{k \times k}$ are constant operators,

$$(4.57) \quad \begin{aligned} A_{lij} &= \langle \phi_i(x) \phi_j'(x), \phi_l(x) \rangle_{\Omega}, \\ B_{lj} &= \nu \langle \phi_j'(x), \phi_l'(x) \rangle_{\Omega}. \end{aligned}$$

Here $\langle \cdot, \cdot \rangle_{\Omega}$ denotes the inner product of two vectors.

In the simulation we set $\nu = 10^{-3}$. The full model is obtained through $n = 2000$ grid points while the trial solution is obtained by extracting the first 10 Fourier modes from a coarse simulation with $k = 100$ grid points. Since one-dimensional Burgers equation has a positive velocity, a wave will propagate to the right with the higher velocities overcoming the lower velocities and creating steep gradients. This steepening continues until a balance with dissipation is reached, which is shown through the velocity profiles at $t = 0.5$ in Figure 4.4(a). Since the solution of Burgers equation shows a high variability with time evolution, more modes are needed in order to present all the snapshots with high accuracy. Meanwhile, IRPI require more iterations than the linear equations to obtain the convergence. Equation (2.20) gives adaptive dimension k at each iteration, they are 30, 59 and 92 for the first three iterations. The convergence plots for IRPI is shown in Figure 4.4(b). In the first few iterations, the error of the approximating solution decreases, and then converges to a fixed value, which is mainly determined by the truncation error of the SVD. In order

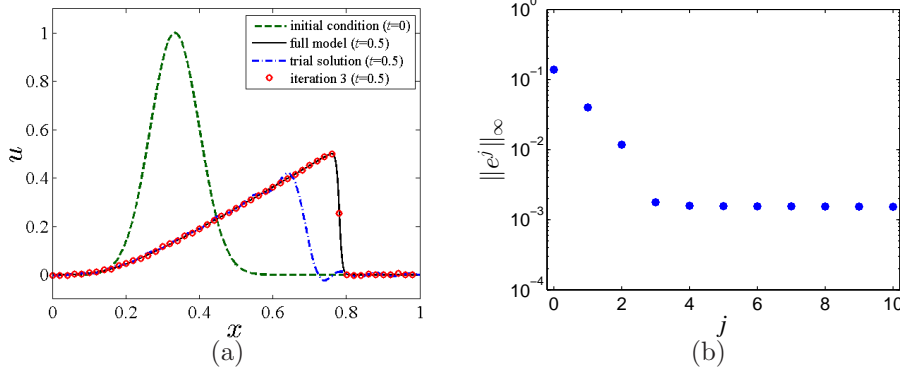


FIGURE 4.4. (Color online.) (a) The velocity profiles at $t = 0$ and $t = 0.5$ of the one-dimensional Burgers equation with constant diffusion coefficient $\nu = 10^{-3}$. $N = 2000$ grid points are used to obtain the full model for the fixed space domain $[0, 1]$ while $n = 100$ grids are used to obtain the coarse model. The first 10 Fourier modes are extracted to construct the coarse solution. The first three iterations respectively take $n = 30$, $n = 59$ and $n = 92$ dominant modes. (b) Convergence of IRPI. Plot of the maximal difference between the benchmark solution $u(t)$ and the iterative solution $\hat{u}^j(t)$, $\|e^j\|_\infty = \sup\{\|\hat{u}^j(t) - u(t)\|_\infty : t \in [0, 0.5]\}$.

to obtain higher resolution, for each iteration, we need more snapshots to construct the information matrix and keep more modes for the associated reduced model.

In the rest of this subsection, we illustrate that IRPI can serve as an efficient posterior error estimation of any coarse models or reduced models. We use the difference between the actual solution $u(t)$ and the approximating solution $\hat{u}^0(t)$ as a function of t to indicate the accuracy of a coarse model (or a reduced model),

$$(4.58) \quad \|e^0(t)\| = \|u(t) - \hat{u}^0(t)\|.$$

However, in many applications, the actual solution $u(t)$ is unknown or very expensive to obtain. In this case, we could use IRPI to obtain a better approximating solution $\hat{u}^1(t)$ and use the difference between $\hat{u}^1(t)$ and $\hat{u}^0(t)$ for the error estimation,

$$(4.59) \quad \|\Delta^0(t)\| = \|\hat{u}^1(t) - \hat{u}^0(t)\|.$$

Figures 4.2, 4.3, and 4.4 demonstrate that error of $\hat{u}^1(t)$ is significantly smaller than error of $\hat{u}^0(t)$ for both advection-diffusion equation and Burgers equation. Therefore, we expect in general $\|\Delta^0(t)\|$ can be served as a good posterior estimation for the numerical error of $\|e^0(t)\|$, for both linear problems and nonlinear problems. More generally, the error of iterative solutions $\hat{u}^j(t)$ computed by IRPI,

$$(4.60) \quad \|e^j(t)\| = \|u(t) - \hat{u}^j(t)\|,$$

can be approximated by the difference between $\hat{u}^{j+1}(t)$ and $\hat{u}^j(t)$,

$$(4.61) \quad \|\Delta^j(t)\| = \|\hat{u}^{j+1}(t) - \hat{u}^j(t)\|.$$

For this reason, in algorithm 3, the criterion $\|\hat{x}^j - \hat{x}^{j-1}\|_\infty < \epsilon$ is used to indicate the convergence of IRPI.

Come back to the example of one-dimensional Burgers equation. Figure 4.5 shows $\|\Delta^j(t)\|$ is a good approximation for the actual error $\|e^j(t)\|$ with time evolution for $t \in [0, 0.5]$.

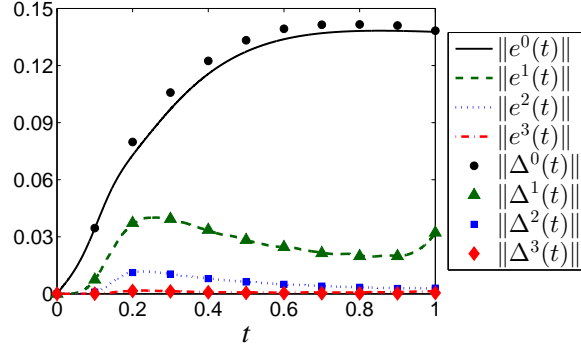


FIGURE 4.5. (Color online.) Comparison of the actual error $\|e^j(t)\| = \|u(t) - \hat{u}^j(t)\|$ with the estimated error $\|\Delta^j(t)\| = \|\hat{u}^{j+1}(t) - \hat{u}^j(t)\|$ for $t \in [0, 0.5]$, where $u(t)$ is the actual solution of the one-dimensional Burgers equation computed by 2000 grid points, $\hat{u}^0(t)$ is the coarse solution computed by 100 grid points and $\hat{u}^j(t)$ is the iterative solution computed by IRPI.

Finally, we should emphasize that IRPI is not an isolated model reduction technique. It can be combined with many existing algorithms to simplify a high dimensional complex system. As an extension, based on IRPI, a local manifold learning approach is discussed in the next section to compute a set of subspaces with much lower dimension where local solutions reside.

5. Locally Linear Projection. Consider a large-scale dynamical system whose solution exhibits vastly different states as it evolves over a large time. In such a case, one might expect that the dominant modes of the system to change significantly overtime. As a result the snapshots at earlier evolution times might not carry much representative information on the state of the system at later times. In this case, two options might be considered: 1) build a global information matrix from snapshots from $t = 0$ to the final time of interest; 2) build the overall solution from a set of smaller information matrices over smaller time segments which each of them span a subspace for a local solution. The classical Galerkin-POD approach takes the first option and the corresponding information matrix often needs a large number snapshots in order to represent the full trajectory and its vector field. Large number of snapshots can give rise to large computation consequently, for two aspects. First of all, the computational cost of SVD of information matrix is proportional to the square of the number of snapshots, as shown in equation (4.46). In addition, a large number of snapshots span a reduced space with a relatively high dimension. Accordingly, more computational cost is needed to form a reduced model. This effect is even considerably increased when treating a nonlinear problem. For example, as in (4.57), the cost to construct the coefficient A_{lij} for the reduced model of Burgers equation is proportional to the cube of the dimension of the reduced space, k . If k is large, the cost for Galerkin projection may even outweigh the cost to solve the original equation. Meanwhile, more modes is computed for each time step, while some of which are redundant for a specified time.

As a consequence, the goal is to provide a reduced modelling technique whose complexity is independent of the time evolution. It is noticed that a manifold embedded in a high dimensional space is locally similar enough to a linear subspace. Among many dimensionality reduction algorithms, Locally Linear Embedding (LLE) [35] might be the most classical one to discover nonlinear structure in high dimensional

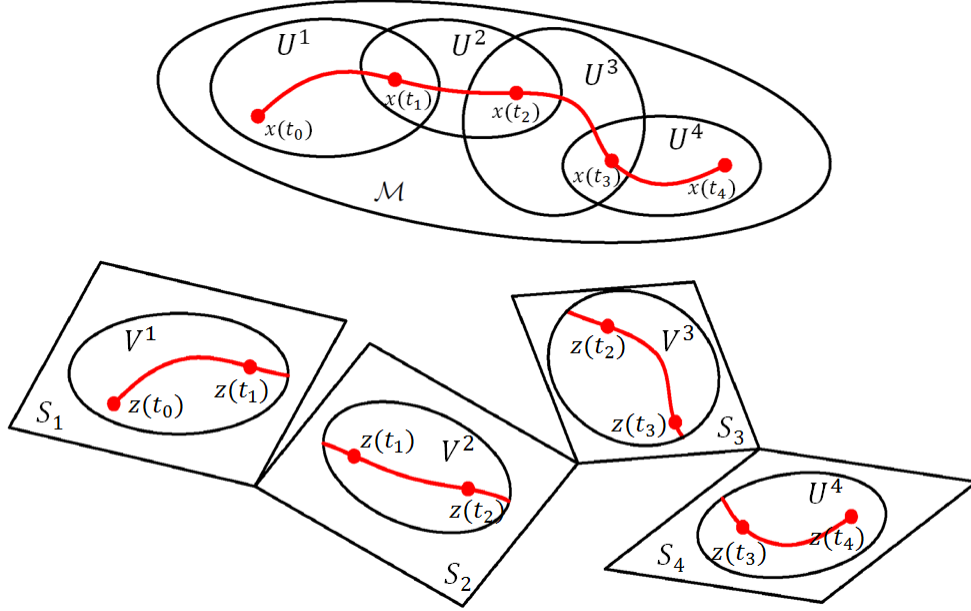


FIGURE 5.1. Graphical description of locally linear projection. $U_i \subset \mathcal{M}$ is an open set that contains the local solution $x(t)$ for the time interval $t \in J_i$. $\varphi_i : U_i \rightarrow V_i$ is the linear map defining the chart. V_i is an open set in a local subspace S_i . The dimension of S_i is significantly less than the dimension of the subspace than contains the entire solution.

data by exploiting the locally linear reconstructions. Since an orbit of the dynamical system (2.1) is a one-dimensional curve, it is possible to use a linear subspace with significant lower dimension $k \geq 2$ to represent the local structure of the curve. The concept of time domain decomposition was already introduced in [37] in order to study fine dynamics by a set of locally invariant manifolds. Adaptive time partition technique is developed in [14]. Moreover, space domain partition [13] and parameter domain partition [15] have been applied in POD method. All of these techniques can sharply reduce the dimension of the original system. In this section, we combine the idea of time domain separation with IRPI and propose a locally linear projection (LLP) algorithm for model reduction. Convergence analysis is also provided. Then we apply this algorithm for the numerical simulation of the Navier-Stokes equation in a cavity flow problem.

5.1. Algorithm of Locally Linear Projection. Suppose $\mathcal{M} \subset \mathbb{R}^n$ is a sub-manifold which contains the solution trajectory $x(t)$ for all $t \in J$. Then in this time domain, by definition of the tangent space, the derivative $\dot{x}(t)$ satisfies $\dot{x}(t) \subset T_{x(t)}\mathcal{M}$, which leads to $f(t, x) \subset T_{x(t)}\mathcal{M}$. Consider an arbitrary $x_0 \in \mathcal{M}$ and a chart (U, φ) with $x_0 \in U$, where $\varphi : U \rightarrow V$ is the map defining the chart. Using the local coordinate $x = \eta(z) = \varphi^{-1}(z, \mathbf{0})$ of the manifold, we can obtain the differential equation in terms of z . By using the chain rule, we have $\dot{x}(t) = \eta'(z(t))\dot{z}(t)$. Then the initial value problem (2.1) becomes

$$(5.1) \quad \dot{z} = \eta'(z(t))^+ f(t, \eta(z)),$$

where $A^+ = (A^T A)^{-1} A$ denotes the pseudo-inverse of a matrix with full column rank. The initial condition z_0 is given by $x_0 = \eta(z_0)$. The existence and uniqueness of the

solution in (5.1) has been proved in [19]. The solution of (2.1) and (5.1) are related by $x(t) = \eta(z(t))$. So any approximation of $z(t)$ provides an approximation for $x(t)$.

In this section, the whole time domain $J := [0, T]$ is partitioned into M smaller subintervals J_1, \dots, J_M with $J_i := [t_{i-1}, t_i]$. We slightly abuse the notation and denote by the subscript i the subinterval index. Let $t_0 = 0$ and $t_M = T$, so that $J = \cup_{i=1}^M J_i$. Let $U_i \subset \mathcal{M}$ be an open set that contains the local solution $x(t)$ for $t \in J_i$. For simplicity, we can use a linear map $\varphi_i : U_i \rightarrow V_i$ to construct a chart (U_i, φ_i) , as Figure 5.2. Let $k = \dim(V_i)$, be the dimension of local coordinate z . In this case, $\eta(z(t)) = \Phi_i z(t)$ for $t \in J_i$, where $\Phi_i \in \mathbb{R}^{k \times n}$. When Φ is formed by k ectohomonal basis vectors, (5.1) degenerate to the locally reduced model formed by the Galerkin projection, i.e.,

$$(5.2) \quad \dot{z} = \Phi_i^T f(t, \Phi_i z),$$

for $t \in J_i$. Then IRPI can be applied to find a good approximation of the invariant subspace and obtain a convergent solution. Specifically, for each subinterval, we set a trial solution, $\hat{x}^0(t)$ initially. During iteration j an extended data ensemble, \hat{Y}_i^{j-1} , containing a few snapshots within the subinterval, is constructed from an approximating solution trajectory and then served for generating the empirical eigenfunctions of a subspace to be used in the next iteration cycle. After locally projecting the full model onto this subspace and constructing a reduced model through (5.2), the time integration of Picard iteration is carried out implicitly to obtain an updated approximation. After convergence, one can move forward to the next subinterval.

Suppose a convergent solution $\hat{x}(t)$ for subinterval i is obtained by IRPI. Let $\xi_i := \hat{x}(t_{i-1})$ be the starting state of this subinterval and $\zeta_i := \hat{x}(t_i)$ be the ending state. The initial sate ξ_i for subinterval i is given by

$$(5.3) \quad \xi_i = \begin{cases} x_0 & \text{if } i = 1, \\ \zeta_{i-1} & \text{if } i > 1. \end{cases}$$

There are several options to estimate the trial solution $\hat{x}^0(t)$ for $t \in J_i$, and we just list a few here. One can simply set the trial solution for subinterval i as a constant, which means $\hat{x}^0(t) = \xi_i$ for $t \in J_i$ (although it is very inaccurate). Alternatively, a coarse model can be used to obtain a rough estimation of $\hat{x}_i(t)$. These two methods can also be used for IRPI, as discussed in the previous section. The third option is to use the time history of solution trajectory to obtain an initial estimation of the invariant subspace. Similar to [26, 29], one can assume the solution for the subinterval i approximately reside in the invariant subspace of the previous one. Thus, a set of empirical eigenfunctions can be generated by SVD of the solution matrix or information matrix by snapshots in J_{i-1} . The general form of the extended data ensemble is

$$(5.4) \quad Y_i^0 = [\hat{x}(t), \gamma f(\hat{x}(t))],$$

for all $t \in J_{i-1}$. Especially, if only the starting snapshot and the ending snapshot are used to construct the discretized information matrix, we have

$$(5.5) \quad \hat{Y}_i^0 = [\zeta_{i-2}, \zeta_{i-1}, \gamma f(t_{i-2}, \zeta_{i-2}), \gamma f(t_{i-1}, \zeta_{i-1})].$$

After projecting the full model onto this subspace, we can calculate the trial solution for $t \in J_i$. Since time-history-based initialization can not be used for the first

subinterval, therefore, we have to combine it with one of the other methods for initialization.

After obtaining a trial solution for a subinterval, IRPI is used to obtain a better approximation of the actual solution. For iteration j , the information matrix is given by

$$(5.6) \quad Y_i^j = [\hat{x}^j(t), \gamma f(\hat{x}^j(t))],$$

for all $t \in J_i$. It is noticed that the reduced subspace of a local solution has a significantly lower dimension when the width of the corresponding subinterval is small enough. In this case, only a small number of snapshots m is needed from the trajectory of the previous iteration. For example, if $m = 2$, the information matrix \hat{Y}_i^j can be constructed from snapshot t_{i-1} and t_i ,

$$(5.7) \quad \hat{Y}_i^j = [\zeta_{i-1}, \hat{x}^j(t_i), f(t_{i-1}, \zeta_{i-1}), f(t_i, \hat{x}^j(t_i))].$$

For the next step, POD (SVD) can be used to construct the empirical eigenvalues Φ_i^j for the reduced subspace. Theoretically the dimension of the reduced subspace can be lower than $2m$. However, as long as m is small enough, say $m = 2$, there is no need to apply POD for further dimension reduction here. Instead, Φ_i^j can be computed more efficiently by the Gram-Schmidt process. Then the full model is projected onto the subspace spanned by Φ_i^j , and a local reduced model is obtained. The algorithm 4 represents the complete process of LLP.

5.2. Convergence Analysis. We have already shown the capability of IRPI to effectively find a global invariant subspace for a dynamical system, and thus generate a sequence of functions that converges to the actual solution. Since LLP is an extension of IRPI, a convergent solution can be obtained for each subinterval, and a combination of these local solutions provides a full trajectory for the original system.

Begin with the first subinterval. In an ideal situation, $x_0 \in S_1^j$, while the solution and vector field satisfy $\hat{x}^{j-1}(t) \subset S_1^j$ and $f(t, \hat{x}^{j-1}) \subset S_1^j$ for all $t \in J_1$, respectively. As theorem 4.7 indicates, LLP can obtain the convergent solution for $t \in J_1$. If the vector field in the initial value problem (2.1) is Lipschitz, then the solution $x(t) = w(t; x(t_1))$ continually depends on the initial condition $x(t_1)$. For this reason, starting from ξ_2 , we can obtain a sequence of functions that converges to $x(t)$ for $t \in J_2$. We can move forward to the rest of subintervals, and achieve the following theorem.

THEOREM 5.1 (Convergence of LLP). *Consider solving the initial value problem (2.1) by LLP for the time domain $J = [0, T]$, which is partitioned into M smaller subintervals J_1, \dots, J_M with $J_i := [t_{i-1}, t_i]$. Let $t_0 = 0$ and $t_M = T$, so that $J = \cup_{i=1}^M J_i$. Suppose the vector field $f(t, x)$ is locally Lipschitz on D' , where D' is an open set contains $(t, x(t))$ for all $t \in J$. For subinterval i , the IRPI approach is applied to obtain an approximation for local solution. Let $x(t)$ be the local solution of the full model, and $\hat{x}^j(t)$ be the solution of the reduced model at iteration j . In an ideal situation, $\xi_i \in S_i^j$, where ξ_i , the initial condition of subinterval i , is given by (5.3). Furthermore, the solution and vector field satisfy $\hat{x}^{j-1}(t) \subset S_i^j$ and $f(t, \hat{x}^{j-1}) \subset S_i^j$ for all $t \in J_{i-1}$, respectively. Then for all $i \in \{1, \dots, M\}$ and $t \in J_i$, the sequence of functions $\hat{x}^j(t)$ uniformly converges to $x(t)$ as $j \rightarrow \infty$.*

Finally, it is interesting to consider LLP as a generalization of many current time integration schemes. We take $m = 2$ as an example. The time-history-initialization

Algorithm 4 Locally linear projection (LLP)**Require:** The initial value problem (2.1).**Ensure:** An approximating solution $\hat{x}(t)$.Divide the whole time domain into smaller subintervals J_1, \dots, J_M **for** subinterval i **do**Set the initial condition (5.3) and obtain a test function $\hat{x}^0(t)$ as the trial solution.
Initialize the iteration number $j = 0$.**repeat**1: Update the iteration number $j = j + 1$.2: Assemble snapshots of an approximating solution $\hat{x}^{j-1}(t)$ into matrix form \hat{X}_i^{j-1} .3: Compute vector field matrix \hat{F}_i^{j-1} whose columns associated with snapshots in \hat{X}_i^{j-1} .4: Form an information matrix for the extended data ensemble $\hat{Y}_i^{j-1} = [\hat{X}_i^{j-1}, \gamma \hat{F}_i^{j-1}]$.5: Based on \hat{Y}_i^{j-1} , compute the empirical eigenfunctions Φ_i^j through Gram-Schmidt process.6: Project the original equation onto a linear subspace spanned by Φ_i^j and form a reduced model (4.2).7: Solve the reduced model and obtain an approximating solution $z_i^j(t)$ in the subspace coordinate system.8: Express the updated solution in the original coordinate system $\hat{x}_i^j(t) = \Phi_i^j z_i^j(t)$.**until** $\|\hat{x}^j - \hat{x}^{j-1}\|_\infty < \epsilon$, where ϵ is the tolerant error.Obtain the local solution $\hat{x}(t) = \hat{x}^j(t)$, for $t \in J_i$.**end for**

provides a linear subspace spanned by \hat{Y}_i^0 to estimate the trial solution for $t \in J_i$. Especially, when $t = t_i$, the initial estimation of the state vector is given by

$$(5.8) \quad \hat{x}^0(t_i) = \hat{Y}_i^0 \cdot \varsigma_i^0,$$

where \hat{Y}_i^0 is given by (5.5) and ς_i^0 is a vector that contains 4 elements. Let δT denote the average width of one subinterval, $\delta T := (T - t_i)/M$; and δt denote the width of one step for the time integration. In the limit when $\delta T = \delta t$, LLP degenerates to the two-step Adams-Bashforth scheme if

$$\varsigma_i^0 = \left[0, 1, -\frac{1}{2\gamma\delta t}, -\frac{3}{2\gamma\delta t} \right]^T.$$

On the other hand, if one uses the IRPI technique to obtain a better estimation at $t = t_i$, the approximating solution is given by

$$(5.9) \quad \hat{x}^1(t_i) = \hat{Y}_i^1 \cdot \varsigma_i^1,$$

where it is assumed only two snapshots are used to construct the information matrix \hat{Y}_i^1 , as expressed by (5.7). In the limit when $\delta T = \delta t$, LLP degenerates to the Crank-Nicholson scheme if

$$\varsigma_i^1 = \left[1, 0, \frac{1}{2\gamma\delta t}, \frac{1}{2\gamma\delta t} \right]^T.$$

More generally, assume n snapshots are sampled from the previous subinterval, then in the limit $\delta T = \delta t$, the time-history initialization can degenerate to the n -step Adams-Bashforth method if proper coefficients are set for ς_i^0 . In addition, if the Y_i^j has $n+1$ snapshots from J_i , and the first n snapshots are overlapping with J_{i-1} , then each iteration defined by IRPI can degenerate to the n -step Adams-Moulton method. Furthermore, if $\delta T = n\delta t$, then each iteration defined by IRPI is a generalized form of the n -order Runge-Kutta method with variable coefficients.

However, as a manifold learning technique, LLP determines a set of eigenfunctions for a subspace and then use a reduced model to calculate the coefficient values for the next subinterval. This is more flexible than a common scheme for the time integration since the later uses predesigned coefficients for each vectors in the information matrix. Therefore, LLP has the ability to provide more stable results for a fixed time interval. Even if $\delta T \gg \delta t$, LLP can still obtain stable results with a high accuracy. In the next subsection, the LLP approach is applied to a lid-driven cavity flow problem.

5.3. Cavity Flow Problem. Consider a lid-driven cavity flow problem in a rectangular domain $\Omega = [0, 1] \times [0, 1]$. The space domain is fixed in time. Mathematically the problem can be represented in terms of the stream function ψ and vorticity ω formulation of the incompressible Navier-Stokes equation. In non-dimensional form, the governing equations are given as

$$(5.10) \quad \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega,$$

$$(5.11) \quad \frac{\partial \omega}{\partial t} = -\frac{\partial \psi}{\partial y} \frac{\partial \omega}{\partial x} + \frac{\partial \psi}{\partial x} \frac{\partial \omega}{\partial y} + \frac{1}{\text{Re}} \left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2} \right),$$

where, Re is the Reynolds number, and x and y are the Cartesian coordinates. The velocity field is given by $u = \partial \psi / \partial y$, $v = -\partial \psi / \partial x$. The zero-slip condition at the nonporous walls yields the following boundary conditions for ψ ,

$$(5.12) \quad \psi_B = 0,$$

where B denotes the points on the walls. The boundary conditions for ω are derived from the definition of ω as given by (5.10). The first-order-correct boundary conditions are given by

$$(5.13) \quad \omega_B = (\psi_{B-1} - \psi_{B-2}) \frac{2}{h^2} + \left(\frac{\partial \psi}{\partial n} \right)_B \frac{2}{h} + O(h).$$

where $B-i$ denotes the interior points whose distance from the boundary equals $i \times h$, and $(\partial \psi / \partial n)_B$ is the tangent velocity of the boundary. At $y = 1$, $(\partial \psi / \partial n)_B = 1$. $(\partial \psi / \partial n)_B = 0$ on other walls. The initial condition is set as $u(x, y) = v(x, y) = 0$. The discretization is performed on a uniform mesh with a second-order central finite difference approximations for second-order derivatives in (5.10) and (5.11). The convective term in (5.11) is discretized via a first-order upwind difference scheme. For the time integration of (5.11), the implicit Crank-Nicolson scheme is applied for the diffusion term, and the explicit two-step Adams-Bashforth method is employed for the advection term.

In the numerical simulation, Reynolds number is set as $\text{Re} = 1000$. The full model uses 129×129 grid points with $\delta t = 5 \times 10^{-3}$ as the unit time step. The whole time domain, $[0, 50]$, is divided into 250 subintervals. For each subinterval, the trial solution

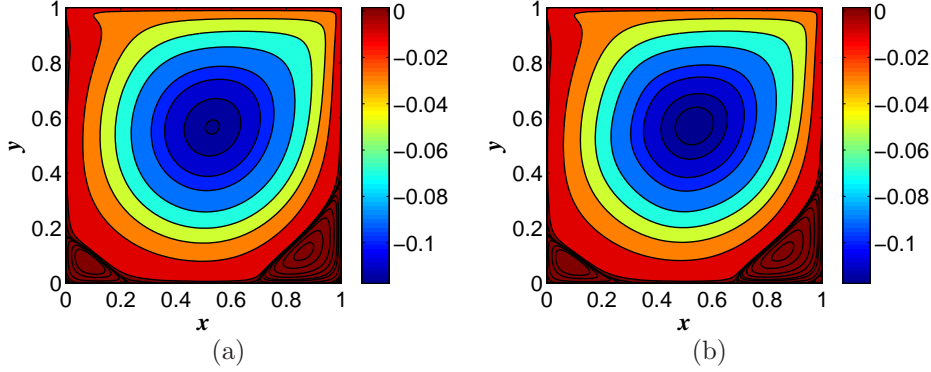


FIGURE 5.2. Streamline pattern for driven cavity problem with $Re=1000$. (a) The full model uses 129×129 grid points. (b) The approximating result obtained through LLP. The whole time domain, $[0, 50]$, is partitioned into 250 subintervals. For each subinterval, a trial solution is calculated from 33×33 grid points. An average of 5 iterations are used to achieve a better approximation. We plot the contours of ψ whose values are -1×10^{-10} , -1×10^{-7} , -1×10^{-5} , -1×10^{-4} , -0.01 , -0.03 , -0.05 , -0.07 , -0.09 , -0.1 , -0.11 , -0.115 , -0.1175 , 1×10^{-8} , 1×10^{-7} , 1×10^{-6} , 1×10^{-5} , 5×10^{-5} , 1×10^{-4} , 2.5×10^{-4} , 1×10^{-3} , 1.3×10^{-3} , and 3×10^{-3} .

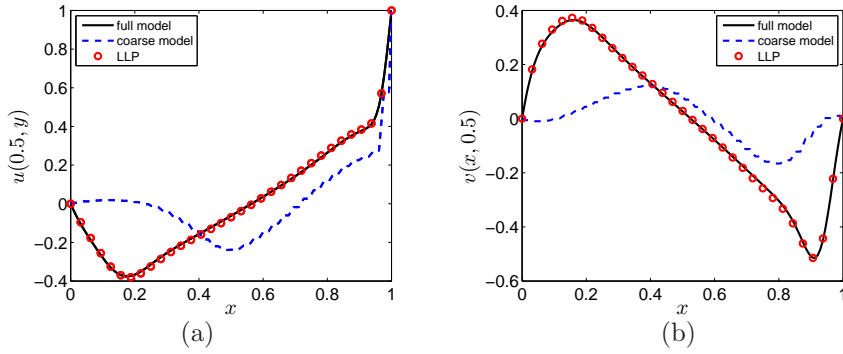


FIGURE 5.3. (a) Comparison of velocity component $u(x=0.5, y)$ along the y -direction passing geometric center between full model and reduced models at $t=50$. (b) Comparison of velocity component $v(x, y=0.5)$ along the x -direction passing geometric center between full model and reduced models at $t=50$. Velocity profiles based on the coarse model, which uses 33×33 grid points, significantly deviate from the actual ones. For each subinterval, LLP uses the coarse model for the trial solution, and then use the IRPI approach to obtain a better local solution with high accuracy.

is obtained through a simulation based on 33×33 coarse grid points with $4\delta t$ as the unit time step. A sequence of functions defined by LLP is used to approach the local solution. For each iteration, 3 snapshots are taken to span the local subspace. Instead of SVD, Gram-Schmidt process is used to form the local empirical eigenfunctions. An average of 5 iterations is carried out to obtain a local convergent solution for each subinterval.

The streamline contours for the lid-driven cavity flow are shown in Figure 5.2. In 5.2(a), the full model matches well with the simulation results from [16], and the values of ψ of the contours are the same as table III of [16]. LLP provides an approximating solution. The main error occurs around the vortex center, where the contour of $\psi = -0.1175$ is missing. Figure 5.3 shows the velocity profiles for u along

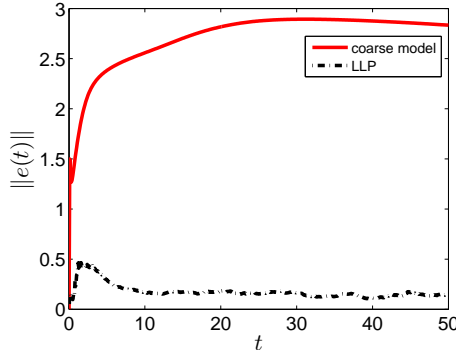


FIGURE 5.4. Plot of the time evolution of the vorticity error. The solid (red) line shows the norm-2 difference between the benchmark solution $\omega(t)$ and the coarse solution $\hat{\omega}^0(t)$. $N = 129 \times 129$ grids are used to obtain the full model with time step $\delta t = 5 \times 10^{-3}$. The coarse model uses $n = 33 \times 33$ grids with time step $4\delta t$. The dashed line shows the norm-2 difference between the $\omega(t)$ and the approximating solution $\hat{\omega}(t)$ obtained by LLP. An average of 5 iterations are used to obtain the convergence for all the subintervals.

vertical lines and v along horizontal lines passing through the geometric center of the cavity. We compare the approximating solution obtained by LLP with the coarse results which are obtained by the 33×33 grid points. Starting from the coarse solution, LLP is able to obtain a much more accurate results for $u(x = 0.5, y)$ and $v(x, y = 0.5)$. Finally, Figure 5.4 shows the L2 error for the time evolution of vorticity for both coarse model and LLP. In terms of vorticity, LLP can significantly reduce the error from the coarse model with an average of 5 iterations.

6. Conclusion. In this article, a new online manifold learning framework, implicitly reduced Picard iteration (IRPI), is proposed for reduced-order modeling of large-scale nonlinear problems where both the data sets and the dynamics are systematically reduced. This framework does not require prior simulations or experiments to obtain data vectors. During each iteration cycle, an approximating solution is calculated in a low dimensional subspace, and provides many snapshots to construct an information matrix. A POD (SVD) method can be applied to generate a set of empirical eigenfunctions which span a new subspace. In an ideal case, a sequence of functions defined by IRPI uniformly converges to the actual solution of the original problem. We also provide an error bound for IRPI, considering the truncation error. The capability of IRPI to solve a high dimensional system with high accuracy is demonstrated in both linear advection-diffusion equation and nonlinear Burgers equation. Moreover, IRPI can also be used as an efficient posterior error estimator for any other coarse models or reduced models.

In order to reduce the cost of the IRPI approach, the LLP technique is developed as an extension of IRPI. Using the idea of partitioning the time domain, the IRPI technique is used to obtain a better approximation for each subinterval. Since each subinterval has much less solution variations, the associated subspace has significantly lower dimension than the overall problem over the entire time domain. The numerical results of nonlinear Navier-Stokes equation through a cavity flow problem imply that the LLP can obtain significant speedups while maintaining a good accuracy.

There are still many other related questions to study in future. For example, since the choice of the extended data ensemble is not unique, there might be other methods

to form the information matrix which results in more efficiently reduced model. It is noticed that POD (SVD) is not the unique technique to find the dominant modes based on the information matrix defined in this paper. How to combine IRPI with other fast model reduction techniques remains a topic for future research.

REFERENCES

- [1] V. ALGAZI AND D. SAKRISON, *On the optimality of Karhunen-Loève expansion*, IEEE Trans. Inform. Theory, 15 (1969), pp. 319–321.
- [2] M. AMABILI, A. SARKAR, AND M. P. PAIDOUSSIS, *Reduced-order models for nonlinear vibrations of cylindrical shells via the proper orthogonal decomposition method*, Journal of Fluids and Structures, 18 (2003), pp. 227–250.
- [3] A. C. ANTOUNAS, D. C. SORESENSEN, AND S. GUGERCIN, *A survey of model reduction methods for large-scale systems*, Contemporary Mathematics, 280 (2001), pp. 193–219.
- [4] P. ASTRID, S. WEILAND, K. WILLCOX, AND T. BACKX, *Missing point estimation in models described by proper orthogonal decomposition*, IEEE Trans. Automat. Control, 53 (2008), pp. 2237–2251.
- [5] J. A. ATWELL, J. T. BORGGAARD, AND B. B. KING, *Reduced order controllers for Burgers’ equation with a nonlinear observer*, Int. J. Appl. Math. Comput. Sci., 11 (2001), pp. 1311–1330.
- [6] Z. BAI, *Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems*, Appl. Numer. Math., 43 (2002), pp. 9–44.
- [7] Z. BAI AND Y. SU, *Dimension reduction of second-order dynamical systems via a second-order Arnoldi method*, SIAM J. SCI. COMPUT., 25 (2005), pp. 1692–1709.
- [8] M. BARRAULT, Y. MADAY, N. C. NGUYEN, AND A. T. PATERA, *An ‘empirical interpolation’ method: Application to efficient reduced-basis discretization of partial differential equations*, C. R. Acad. Sci. Paris, Ser. I, 339 (2004), pp. 667–672.
- [9] M. BELKIN AND P. NIYOGI, *Laplacian eigenmaps for dimensionality reduction and data representation*, Journal Neural Computation, 15 (2003), pp. 1373–1396.
- [10] M. BERGMANN, L. CORDIER, AND J. P. BRANCHER, *Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced-order model*, Physics of Fluids, 17 (2005), pp. 80–101.
- [11] C. M. BISHOP, *Pattern Recognition and Machine Learning*, Springer, New York, 2011.
- [12] S. CHATURANTABUT AND D. C. SORESENSEN, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 32 (2010), pp. 2737–2764.
- [13] C. F. D. AMSALLEM, M. J. ZAHR, *Nonlinear model order reduction based on local reduced-order bases*, in press in the International Journal for Numerical Methods in Engineering, (2012). available online.
- [14] M. DIHLMANN, M. DROHMANN, AND B. HAASDONK, *Model reduction of parametrized evolution problems using the reduced basis method with adaptive time partitioning*, Technical Report 2011-13, Stuttgart Research Centre for Simulation Technology, Stuttgart, Germany, May 2011.
- [15] J. L. EFTANG, A. T. PATERA, AND E. M. R. NQUIST, *An “hp” certified reduced basis method for parametrized elliptic partial differential equations*, SIAM J. Sci. Comput., 32 (2010), pp. 3170–3200.
- [16] U. GHIA, K. N. GHIA, AND C. T. SHIN, *High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method*, Journal of Computational Physics, 48 (1982), pp. 387–411.
- [17] M. GRAHAM AND I. KEVREKIDIS, *Alternative approaches to the Karhunen-Loève decomposition for model reduction and data analysis*, Computers and Chemical Engineering, 20 (1996), pp. 495–506.
- [18] M. A. GREPL, Y. MADAY, N. C. NGUYEN, AND A. T. PATERA, *Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations*, M2AN Math. Model. Numer. Anal., 41 (2007), pp. 575–605.
- [19] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, Springer, New York, 2006.
- [20] P. HOLMES, J. LUMLEY, AND G. BERKOOZ, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, Cambridge Univ. Press, Cambridge, UK, 1998.
- [21] N. KAMBHATLA AND T. K. LEEN, *Dimension reduction by local principal component analysis*, Neural Computation, 9 (1997), pp. 1493–1516.

- [22] S. LALL, J. E. MARSDEN, AND S. GLAVAŠKI, *A subspace approach to balanced truncation for model reduction of nonlinear control systems*, Int. J. Robust Nonlin. Contr., 12 (2002), pp. 519–535.
- [23] J. A. LEE AND M. VERLEYSEN, *Nonlinear Dimensionality Reduction*, Springer Verlag, New York, December 2007.
- [24] M. LOÈVE, *Probability Theory*, Van Nostrand, New York, 1955.
- [25] K. V. MARDIA, J. T. KENT, AND J. M. BIBBY, *Multivariate Analysis*, Academic Press, London, February 1980.
- [26] R. MARKOVINOVIĆ AND J. D. JANSEN, *Accelerating iterative solution methods using reduced-order models as solution predictors*, Int. J. Numer. Meth. Engng, 68 (2006), pp. 525–541.
- [27] J. D. MEISS, *Differential Dynamical Systems*, SIAM, Philadelphia, 2007.
- [28] P. PARRILO, F. PAGANINI, G. VERGHESE, B. LESIEUTRE, AND J. E. MARSDEN, *Model reduction for analysis of cascading failures in power systems*, in American Control Conference, San Diego, CA, June 1999, pp. 4208–4212.
- [29] M. RAPÚN AND J. M. VEGA, *Truncation error models based on local POD plus Galerkin projection*, Journal of Computational Physics, 229 (2010), pp. 3046–3063.
- [30] M. RATHINAM AND L. R. PETZOLD, *Dynamic iteration using reduced order models: A method for simulation of large scale modular systems*, SIAM J. Numer. Anal., 40 (2002), pp. 1446–1474.
- [31] ———, *A new look at proper orthogonal decomposition*, SIAM J. Numer. Anal., 41 (2003), pp. 1893–1925.
- [32] M. REWIŃSKI AND J. WHITE, *A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices*, IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 22 (2003), pp. 155–170.
- [33] ———, *Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations*, Linear Algebra Appl., 415 (2006), pp. 426–454.
- [34] A. ROSENFELD AND A. C. KAK, *Digital Picture Processing*, Academic Press, New York, 1982.
- [35] S. ROWEIS AND L. SAUL, *Nonlinear dimensionality reduction by locally linear embedding*, Science, 290 (2000), pp. 2323–2326.
- [36] C. W. ROWLEY, T. COLONIUS, AND R. M. MURRA, *Model reduction for compressible flows using POD and Galerkin projection*, Physica D, 189 (2004), pp. 115–129.
- [37] A. SAWANT AND A. ACHARYA, *Model reduction via parametrized locally invariant manifolds: Some examples*, Computer Methods in Applied Mechanics and Engineering, 195 (2006), pp. 6287–6311.
- [38] J. M. A. SCHERPEN, *Balancing for nonlinear systems*, Systems & Control Letters, 21 (1993), pp. 143–153.
- [39] S. SHVARTSMAN AND I. KEVREKIDIS, *Low-dimensional approximation and control of periodic solutions in spatially extended systems*, Phys. Rev. E, 58 (1998), pp. 361–368.
- [40] J. TENENBAUM, V. DE SILVA, AND J. LANGFORD, *A global geometric framework for nonlinear dimensionality reduction*, Science, 290 (2000), pp. 2319–2323.
- [41] L. N. TREFETHEN AND D. BAU, *Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
- [42] K. ZHOU AND J. DOYLE, *Essentials of robust control*, Prentice Hall, Englewood Cliffs, NJ, 1998.